ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΜΣ Πληροφοριακά Συστήματα και Υπηρεσίες
Τμήμα Μεγάλων Δεδομένων και Αναλυτικής

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Επεξεργασία ροών χωροκειμενικών δεδομένων

Σπυρίδων Κάσδαγλης
Α.Μ.: ΜΕ1927

Επιβλέπων Καθηγητής:
Χρήστος Δουλκερίδης

ΠΕΙΡΑΙΑΣ
ΦΕΒΡΟΥΑΡΙΟΣ 2022

Περίληψη

Smartphones, wearables, health assistants, instagram, facebook, twitter, tic toc· Αυτές είναι μόνο μερικές -πολύ γνωστές- από τις εκατοντάδες συσκευές και εφαρμογές που πλέον απαιτούν τη συλλογή και επεξεργασία χωροκειμενικών (spatiotextual) δεδομένων, ώστε να μπορούν να προσφέρουν την βέλτιστη εμπειρία για τους χρήστες τους. Όπως είναι επόμενο, η χρήση “έξυπνων” συσκευών και αισθητήρων έχει αυξήσει ραγδαία το μέγεθος της πληροφορίας που έχει να κάνει με την τοποθεσία και το κείμενο. Αντίστοιχα, αυξάνονται οι ανάγκες για επεξεργασία και εξαγωγή αποτελεσμάτων σε πραγματικό χρόνο.

Το Spatiotextual Similarity Join συνεχόμενων ροών χωροκειμενικών δεδομένων είναι ένα από τα πλέον σημαντικά και σύγχρονα ερωτήματα που το συναντάμε σε διάφορες εφαρμογές στο σήμερα. Αναφέρεται στην προσπάθεια να επιτευχθεί ζεύξη μεταξύ των αντικειμένων μιας συνεχόμενης ροής δεδομένων που φέρουν γεωγραφική και κειμενική πληροφορία, με ένα σύνολο χωροκειμενικών αντικειμένων αποθηκευμένων σε ένα σύστημα.
Για παράδειγμα, ας υποθέσουμε πως έχουμε μια συνεχόμενη ροή και ένα σύνολο χωροκειμενικών αντικειμένων, και παράλληλα, μας δίνεται μια ακτίνα αναζήτησης και ένα κατώφλι κειμενικής ομοιότητας (threshold). Εμείς θα προσπαθήσουμε να επιστρέψουμε ζευγάρια αντικειμένων από τα δύο προαναφερθέντα σύνολα, τα οποία βρίσκονται σε απόσταση μικρότερη από την δοθείσα ακτίνα, ενώ η κειμενική τους ομοιότητα κρίνεται μεγαλύτερη από το δοθέν threshold.

Γίνεται εύκολα αντιληπτό πως στις μέρες μας το μέγεθος των χωροκειμενικών δεδομένων είναι αυξημένο και οι ανάγκες επεξεργασίας αυτών σε πραγματικό χρόνο υπερβαίνουν τις δυνατότητες που μπορεί να προσφέρει ένα κεντρικοποιημένο σύστημα. Γι’ αυτό κρίθηκε αναγκαία η υλοποίηση ενός συστήματος που να υιοθετεί την κατανεμημένη τοπολογία και τεχνικές παράλληλης εκτέλεσης εργασιών.

Abstract

Smartphones, wearables, health assistants, instagram, facebook, twitter, tic toc· are only some -well known- of hundreds of devices and applications that nowdays require the collection and processing of spatiotextual data in order to provide the best possible user experience. Therefore, the use of smart devices and sensors has caused a rapid increase of the amount of data that contain location and text info. As a result, processing and real-time result extraction needs rise accordingly.

Spatiotextual similarity join of streaming data is one of the foremost operations in spatiotextual data integration and finds usage in various applications. It refers to the execution of the needed operations in order to achieve join between the streaming data that contain spatiotextual info and a set of spatiotextual objects.
For example, let's assume a set of streaming and a set of static spatiotextual data, as well as a given spatial range radius and a text similarity threshold. We are attempting to determine all the similar pairs from the two sets that are in a closer distance than the given radius, while at the same time their textual similarity ranking is greater than the given threshold.

It is easily understood that nowadays, the big volume of spatiotextual data and the needs for real-time processing go beyond the possibilities that a centralized system can offer. Therefore, the development of a system with decentralized topology which makes use of parallel processing techniques is considered necessary.

Περίληψη **3**

Abstract **4**

1. Εισαγωγή **7**
1.1 Εφαρμογή και απαιτήσεις 7
1.2 Επεξεργασία ροής - Stream Processing 8
1.3 Παρουσίαση προβλήματος και τεχνικών επίλυσης 9
1.4 Δομή Εργασίας 10

2. Σχετικές εργασίες χωροκειμενικών ερωτημάτων **11**
2.1 Τύποι Χωροκειμενικών ερωτημάτων 11
Spatio Textual Boolean AND Query 12
Spatio Textual Boolean OR Query 12
Spatio Textual kNN Query 13
Spatio Textual Top-k Query 13
Spatio Textual Similarity Join 13
2.2 Τεχνικές και Τεχνολογίες Επεξεργασίας Μεγάλων Δεδομένων 14
Κεντρικοποιημένα συστήματα 14
Κατανεμημένα Συστήματα 14
Batch Processing 15
MapReduce 15
Stream Processing 18
2.3 Τεχνικές Επεξεργασίας Συνεχόμενων Χωροκειμενικών Ροών 19
Related Work Τεχνικών και Μεθόδων 19
Σχόλια και παρατηρήσεις 20
Spatial-first και Text-First Indexing 21

3. Ορισμός του προβλήματος **22**
3.1 Παρουσίαση και βασική λογική επίλυσης 22
3.2 Ανάγκες και προκλήσεις 22
3.3 Σχεδιασμός και αντιμετώπιση προβλήματος 23
Spatial Join using Haversine formula 24
Textual Join using Jaccard Similarity 24
3.4 Στόχος Συστήματος 25

4. Μεθοδολογία και Ανάλυση Συστήματος **26**
4.1 Παρουσίαση συστήματος 26
4.2 Εργαλεία και Τεχνολογίες 28
4.2.1 Apache Spark Streaming 28
4.2.2 Twitter Stream API 29

4.3 Partitioning 29
4.3.1 Δημιουργία του Uniform Grid 29
4.3.2 Διεξαγωγή ερωτημάτων και Duplication 32
4.4 Υλοποίηση Συστήματος και Εκτέλεση 36
4.4.1 Configuration Spark Streaming 36
4.4.2 Preprocess and Mapping 36
4.4.3 Reduce and Distribution 37
5. Πειραματική διαδικασία και Αποτελέσματα 39
5.1 Μετρικές αξιολόγησης συστήματος 39
Αντικείμενα ανά Κελί (Objects per Cell) 39
Πλήθος Αντιγράφων 39
Χρόνος εκτέλεσης 39
Πλήθος συγκρίσεων 39
5.2 Datasets 39
5.2.1 Synthetic Datasets 40
Schema 40
Distribution 40
5.2.2 Real Time Stream 41
Schema 41
5.3 Πειραματική Μελέτη 42
5.3.1 Πείραμα 1 42
Αντικείμενα ανά Κελί 43
Πλήθος Αντιγραφών 43
5.3.2 Πείραμα 2 45
Αντικείμενα ανά Κελί 45
Πλήθος Αντιγραφών 46
5.3.3 Πείραμα 3 48
Αντικείμενα ανά Κελί 48
Πλήθος Αντιγραφών 49
5.3.4 Πειραμα 4 51
Αντικείμενα ανά Κελί 52
Πλήθος Αντιγραφών 52
5.3.5 Αποτελέσματα 54
6. Συμπεράσματα και σκέψεις για μελλοντική μελέτη 57
Βιβλιογραφία 58

1. Εισαγωγή

1.1 Εφαρμογή και απαιτήσεις

Δεν υπάρχει αμφιβολία πως ζούμε στην εποχή της πληροφορίας. Η συλλογή (collection), μετασχηματισμός (transformation), αποθήκευση (store), εξαγωγή και απεικόνιση (visualization) αποτελεσμάτων αποτελούν δραστηριότητες των περισσότερων οργανισμών και υπηρεσιών με τους οποίους αλληλοεπιδρούμε στην καθημερινότητα μας.

Η ραγδαία αύξηση του όγκου δεδομένων τόσο με κειμενική (textual) όσο και χωρική (spatial) ταυτότητα συνεχίζει να γίνεται αισθητή ολοένα και περισσότερο τα τελευταία χρόνια, κάτι το οποίο γίνεται εύκολα κατανοητό αν συλλογιστούμε το πλήθος των συσκευών (hardware) αλλά και των εφαρμογών (apps) που κάνουν χρήση των δεδομένων αυτών. Συσκευές όπως: smartphones, wearables, smart home devices, weather trackers, health and medical assistants πλέον απαιτούν την συλλογή και επεξεργασία των χωροκειμενικών (spatiotextual) δεδομένων του χρήστη για την σωστή λειτουργία τους. Παράλληλα, οι περισσότερες εφαρμογές κοινωνικής δικτύωσης (social media), όπως instagram, facebook, twitter, tic toc κτλ, και άλλες, όπως έξυπνοι βοηθοί (smart assistants), για παράδειγμα οι ευρέως διαδεδομένοι Google Assistant και Alexa, τα σύγχρονα προγράμματα περιήγησης διαδικτύου (web browsers), όπως Google Chrome, Mozilla Firefox, Apple Safari, Opera και Microsoft Edge, εφαρμογές πλοήγησης, όπως Google Maps, MapQuest και Waze, εφαρμογές ψηφιακού εμπορίου και διαφήμισης (digital marketing), όπως Google και Facebook Ads, εφαρμογές βραχυπρόθεσμης μίσθωσης οχήματος (ride-hailing), όπως Uber και Beat· όλες χρησιμοποιούν τα χωροκειμενικά δεδομένα με στόχο τη βέλτιστη και προσαρμοσμένη (customized) προς τον χρήστη παροχή των υπηρεσιών και δυνατοτήτων τους. Φυσικά, χρήση των χωροκειμενικών δεδομένων συναντάμε και σε τομείς (domains) όπως η αεροπλοΐα, η ναυσιπλοΐα καθώς και σε επιβατικές και εμπορικές μεταφορές. Η γρήγορη και ορθή χρήση χωροκειμενικών δεδομένων κρίνεται αναγκαία για την επικοινωνία και την ασφαλή υλοποίηση των δραστηριοτήτων των παραπάνω κλάδων.

Όπως διατυπώθηκε και προηγουμένως χωροκειμενικό δεδομένο ορίζουμε ένα αντικείμενο το οποίο φέρει τόσο χωρική (spatial) όσο κειμενική (textual) πληροφορία. Ένα παράδειγμα χωροκειμενικού αντικειμένου είναι μία ανάρτηση ενός χρήστη σε μία πλατφόρμα κοινωνικής δικτύωσης όπως το Twitter. Μία τέτοια ανάρτηση φέρει, εκτός των άλλων, την κειμενική πληροφορία που κοινοποιεί ο χρήστης καθώς και την γεωγραφική τοποθεσία του ίδιου.

Όπως είναι φυσικό, η αύξηση της λίστας χρήσεων των χωροκειμενικών δεδομένων σημαίνει και μία ανάλογη αύξηση του όγκου και της πολυπλοκότητας αυτών, η οποία με την σειρά της οδηγεί στην αύξηση των απαιτήσεων και της ανάγκης για αποτελεσματική επεξεργασία. Η τρέχουσα κλίμακα χωροκειμενικών δεδομένων δεν μπορεί να αντιμετωπιστεί χρησιμοποιώντας παραδοσιακά, κεντρικά συστήματα, κάτι που επέβαλε την υιοθέτηση τεχνικών της επιστήμης των Μεγάλων Δεδομένων (Big Data) καθώς και την επί τούτω ανάπτυξη και χρήση κατανεμημένων χωροκειμενικών συστημάτων επεξεργασίας. Για την διαχείριση, επεξεργασία και αποθήκευση τέτοιου μεγέθους και τύπου δεδομένων χρησιμοποιούνται τεχνικές όπως το Map Reduce, ευρετηρίαση (indexing), επεξεργασία φυσικής γλώσσας (Natural Language Processing) και εργαλεία όπως τα: Apache Spark, Apache Hadoop, HDFS, Apache Hive.

Στα πλαίσια της παρούσας εργασίας θα κληθούμε να απαντήσουμε σε ερωτήματα Similarity Join Συνεχόμενων Ροών δεδομένων και ενός Batch συνόλου χωροκειμενικών αντικειμένων (Query Objects) κάνοντας χρήση χωρικών και κειμενικών μετρικών. Παράλληλα θα αναπτύξουμε μηχανισμούς partitioning και κατανομής των εργασιών στοχεύοντας στην βελτιστοποίηση της εκμετάλλευσης των πόρων του συστήματος.

1.2 Επεξεργασία ροής - Stream Processing

Η πρόκληση της διαχείρισης μεγάλων δεδομένων εντείνεται σε εφαρμογές όπου απαιτείται η επεξεργασία και ο μετασχηματισμός αυτών σε πραγματικό χρόνο (Real Time Processing). Η επεξεργασία ροών (Stream Processing) είναι η διαδικασία κατά την οποία, ένα συνήθως κατανεμημένο σύστημα λαμβάνει, επεξεργάζεται και εξάγει αποτελέσματα για έναν αριθμό συνεχόμενων ροών δεδομένων. Σε αντίθεση με την επεξεργασία δεδομένων αποθηκευμένων κατά παρτίδες σε κάποιο μέσο ή βάση αποθήκευσης δεδομένων (batch processing), κατά την επεξεργασία συνεχόμενων ροών δεδομένων χρησιμοποιούμε τεχνικές ώστε τα δεδομένα τα οποία λήφθηκαν σε πραγματικό χρόνο να υποστούν την κατάλληλη επεξεργασία και μετασχηματισμό εντός πολύ μικρών χρονικών περιόδων.

Οι βασικές αρμοδιότητες ενός συστήματος επεξεργασίας ροών είναι να διασφαλίζει ότι τα δεδομένα ρέουν αποτελεσματικά, ότι το κόστος υπολογισμού κλιμακώνεται και ότι το σύστημα είναι ανεκτικό σε σφάλματα (fault tolerance). Συνήθως, τα περιβάλλοντα επεξεργασίας ροών δεδομένων χρησιμοποιούν κάποιες λειτουργίες αποθήκευσης κατάστασης (state) ή αποθήκευσης πληροφορίας εντός ορισμένου χρονικού πλαισίου (windowing). Χάρη στα προτερήματα που προσφέρει σε συγκεκριμένες εφαρμογές, αλλά και στην ολοένα αυξανόμενη χρήση των εφαρμογών αυτών, η επεξεργασία συνεχόμενων ροών έχει γνωρίσει τεράστια ανάπτυξη τα τελευταία χρόνια. Ένα από τα βασικά προτερήματα αυτά, είναι το ότι η επεξεργασία επιτυγχάνεται σε μικρότερο όγκο

πληροφορίας σε κάθε βήμα εκτέλεσης, εν αντιθέσει με τα batch συστήματα όπου η επεξεργασία πραγματοποιείται στο σύνολο των δεδομένων.

Η χρήση συστημάτων επεξεργασίας ροών αποτελεί μονόδρομο όταν αδυνατούμε να αποθηκεύσουμε τον συνολικό όγκο της πληροφορίας που θα επεξεργαστούμε, κάτι το οποίο προς όφελός μας, μας οδηγεί και σε συστήματα με λιγότερες απαιτήσεις υλικού (Hardware Requirements). Μπορούμε να παρατηρήσουμε την χρήση συστημάτων επεξεργασίας συνεχόμενων ροών δεδομένων σε εφαρμογές παρακολούθησης γραμμών παραγωγής, έξυπνων συστημάτων υγείας, οχημάτων, ανίχνευσης απάτης (fraud detection) και φυσικά, όπως θα περιγράψουμε εκτενώς στα επόμενα κεφάλαια, εφαρμογές που κάνουν χρήση χωροκειμενικών δεδομένων σε πραγματικό χρόνο. Η βασική ιδιαιτερότητα των παραπάνω συστημάτων είναι η κατανομή των δεδομένων στους πόρους ενός κατανεμημένου συστήματος με στόχο την αποδοτικότερη παράλληλη επεξεργασία.

1.3 Παρουσίαση προβλήματος και τεχνικών επίλυσης

Στη συνέχεια παρουσιάζεται ένα παράδειγμα του προβλήματος του οποίου καλούμαστε να επιλύσουμε στα πλαίσια της παρούσας εργασίας.

Ένα σύνολο επιχειρήσεων εστίασης Q ορίζουν η κάθε μία ξεχωριστά μία χωρική ταυτότητα $q_i \in Q$, δοσμένη με μορφή συντεταγμένων $q.lat$, $q.lon$, και ένα σύνολο keywords $q.text$ που χαρακτηρίζουν τις υπηρεσίες της. Παράλληλα δίνεται μία ακτίνα γεωγραφικής αναζήτησης r. Στην συνέχεια, δεχόμαστε μία συνεχόμενη ροή χωροκειμενικών αντικειμένων S ενός συνόλου χρηστών. Κάθε αντικείμενο $s_i \in S$ φέρει τη χωρική θέση του χρήστη, $s.lat$, $s.lon$, καθώς και ένα σύνολο keywords $s.text$ που στην προκειμένη εκφράζουν τα ενδιαφέροντα του. Αν η θέση του χρήστη βρίσκεται εντός του range που έχει ορίζεται από την ακτίνα r και μιας επιχείρησης, ενώ ταυτόχρονα ικανοποιεί και τις κειμενικές συνθήκες ταύτισης οι οποίες είναι σαφώς ορισμένες χρησιμοποιώντας ένα κατώφλι ομοιότητας (similarity threshold) a, τότε, θα λάβει, σε πραγματικό χρόνο, ενημέρωση για το “ταίριασμα” του με τις επιχειρήσεις αυτές.

Το σύστημα του παραδείγματος το οποίο περιγράφεται παραπάνω και θα αναπτύξουμε στην παρούσα εργασία ακολουθεί την τοπολογία ενός Publish and Subscribe μοντέλου.

Μία συνηθισμένη εφαρμογή του παραπάνω γίνεται σε e-coupon promotion υπηρεσίες που χρησιμοποιούνται από επιχειρήσεις, ώστε να ενημερώσουν και να προσελκύσουν πιθανά ενδιαφερόμενους πελάτες.

Στην παρούσα έρευνα, μελετάμε την επεξεργασία και την αποτελεσματικότερη ζεύξη ομοιότητας (similarity join) σε συνεχόμενες ροές χωροκειμενικών δεδομένων. Η μελέτη, οι πειραματικές διαδικασίες και η ανάπτυξη στοχεύουν στην δημιουργία μιας αποδοτικότερης λύσης για την διαχείριση και την ζεύξης Real-Time χωροκειμενικών δεδομένων (micro-batches) και ερωτημάτων (Queries). Για την επίτευξη αυτού, υλοποιήθηκαν τεχνικές διαμέρισης των δεδομένων (partitioning) με χρήση αλγορίθμων που αναπτύχθηκαν στα πλαίσια αυτής της έρευνας, μέθοδοι αντιγραφής αντικειμένων (Duplication) και τεχνικών MapReduce, με στόχο την ομοιόμορφη κατανομή και την παράλληλη εκτέλεση των εργασιών.

1.4 Δομή Εργασίας

Για την καλύτερη κατανόηση του προβλήματος, των αναγκών, αλλά και των κατευθύνσεων που έχουν προηγηθεί και μας οδήγησαν στην λύση που προτείνουμε στην παρούσα εργασία, θα παραθέσουμε τη σχετική βιβλιογραφία, τις πήγες και τα εργαλεία τα οποία μελετήθηκαν. Στη συνέχεια, θα παρουσιάσουμε τα προβλήματα και τις προκλήσεις που πρέπει να ληφθούν υπόψη κατά τον σχεδιασμό και την υλοποίηση ενός τέτοιου συστήματος, πριν αναπτύξουμε αναλυτικά την αρχιτεκτονική, τις τεχνικές, τα εργαλεία και τους αλγορίθμους που χρησιμοποιήθηκαν για την επίτευξη της τελικής μας λύσης. Έπειτα, θα παραθέσουμε όλα τα πειράματα που διεξήχθησαν συνοδευόμενα από διαγράμματα και αναλυτικές περιγραφές αυτών, φτάνοντας, έτσι, στα τελικά συμπεράσματα για την απόδοση του συστήματος, καθώς και σε κάποιες γενικότερες σκέψεις και συμβουλές για μελλοντικές μελέτες.

Συνοψίζοντας τα παραπάνω,η δομή της παρούσας εργασίας χωρίζεται ως εξής:

- Ενότητα 2: Σχετικές εργασίες χωροκειμενικών ερωτημάτων (Related work)
- Ενότητα 3: Ορισμός του προβλήματος
- Ενότητα 4: Ανάλυση και Μεθοδολογία επίλυσης
- Ενότητα 5: Πειραματική διαδικασία και Αποτελέσματα
- Ενότητα 6: Συμπεράσματα και σκέψεις για μελλοντική μελέτη
- Βιβλιογραφία

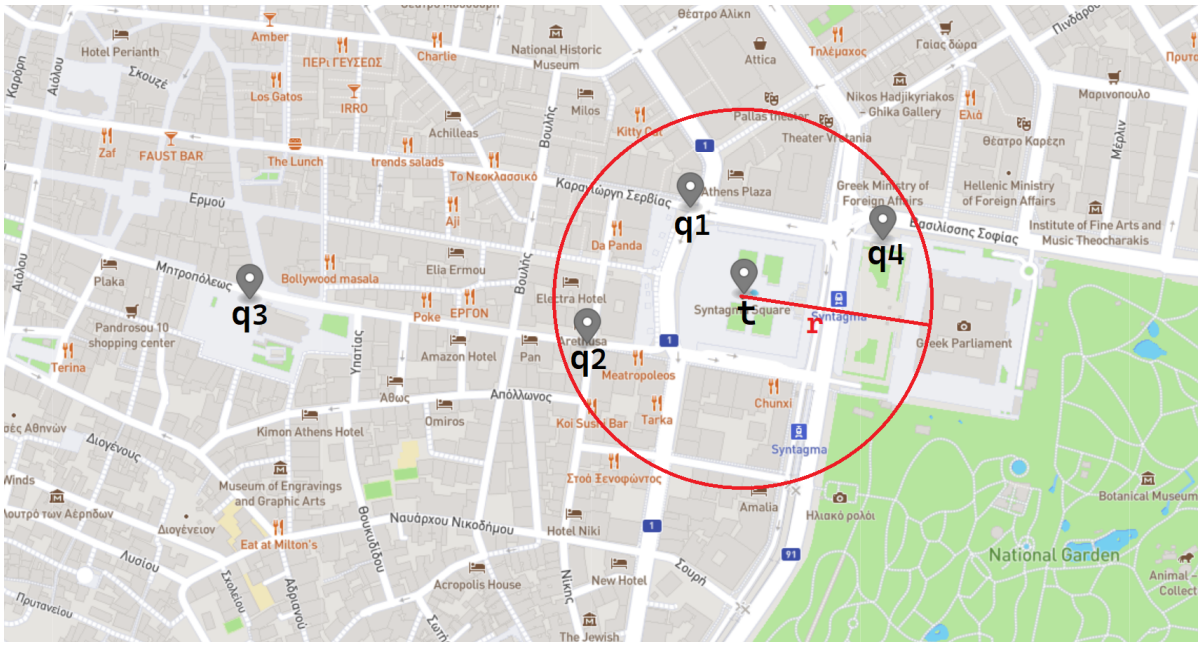
2. Σχετικές εργασίες χωροκειμενικών ερωτημάτων

2.1 Τύποι Χωροκειμενικών ερωτημάτων

Σε αυτή την ενότητα θα αναπτύξουμε τους διαφορετικούς τύπους ερωτημάτων (queries) που εφαρμόζονται σε χωροκειμενικά δεδομένα.

Ορίζουμε ως χωροκειμενικά δεδομένα τα στοιχεία εκείνα που φέρουν γεωγραφίκη και κειμενική πληροφορία. Έτσι, έστω ένα σύνολο δεδομένων $T=\{t_1,t_2,..,t_n\}$, όπου κάθε $t \in T$ περιλαμβάνει μια κειμενική πληροφορία, εκφρασμένη σε λέξεις κλειδιά (keywords), t.text και μια χωρική, ορισμένη με γεωγραφικές συντεταγμένες t.lat, t.lon για το γεωγραφικό πλάτος και γεωγραφικό μήκος αντίστοιχα.

Τα χωροκειμενικά ερωτήματα (spatio textual queries) αποτελούν μία εξέλιξη των spatial ερωτημάτων, εφαρμογή των οποίων μπορούμε να συναντήσουμε σε εργαλεία εύρεσης σημείων ενδιαφέροντος, όπως το Google Maps. Τα τελευταία χρόνια, παρατηρείται ένα έντονο ενδιαφέρον της επιστημονικής κοινότητας για την επεξεργασία χωροκειμένικών ερωτημάτων λόγω του αυξανόμενου αριθμού εφαρμογών τους. Τέτοια είναι το microblogging μέσω πλατφόρμων όπως το Twitter, δεδομένα (tweets) του οποίου χρησιμοποιήσαμε στο πλαίσιο των πειραματικών διαδικασιών της παρούσας έρευνας.



Εικόνα 1 *- Τα σημεία t, q1, q2, q3, q4 και η γεωγραφική εμβέλεια αναζήτησης(ακτίνας r)*

Στην Εικόνα 1 αναπαρίσταται ένα σημείο ενδιαφέροντος (Point of interest - POI) t, η γεωγραφική πληροφορία του οποίου έχει αποδοθεί στο χάρτη και η κειμενική του πληροφορία είναι t.text=[athens, coffee, food, sun]. Παράλληλα, δίνεται μια ακτίνα r που ορίζει την εμβέλεια χωρικής αναζήτησης από το t. Έχουν τοποθετηθεί στον χάρτη και τέσσερα ακόμα query objects (q1,q2,q3,q4) τα οποία φέρουν και κειμενική πληροφορία η οποία περιγράφεται με λέξεις κλειδιά (keywords) ως εξής:

- q1.text=[restaurant, athens]
- q2.text=[athens,coffee,food,sun]
- q3.text=[athens,food,sun]
- q4.text=[coffee,sun]

Παρακάτω παρατίθενται οι τύποι spatio textual queries μαζί με παραδείγματα εφαρμογής αυτών.

➢ Spatio Textual Boolean AND Query

Δοθέντος ενός χωροκειμενικού αντικειμένου t που περιέχει χωρική(t.lat, t.lon) και κειμενική(t.text) πληροφορία, καθώς και μία ακτίνα r, ένα Boolean AND Query επιστρέφει όλα τα χωροκειμενικά query αντικείμενα(Q) τα οποία βρίσκονται εντός της ακτίνας r από το γεωγραφικό στίγμα που εκφράζεται από τις συντεταγμένες t.lat, t.lon, δηλαδή dist({q.lat, q.lon},{t.lat, t.lon})<=r, και περιέχουν όλη την κειμενική πληροφορία t.text, δηλαδή t.text ∈ q.text. Σύμφωνα με αυτό λοιπόν, για τα σημεία τα οποία απεικονίζονται στην Εικόνα 1, το μοναδικό αντικείμενο που επαληθεύει το ερώτημα θα ήταν το q2 καθώς είναι το μόνο εντός απόστασης r από το t που επιτυγχάνει απόλυτη κειμενική ομοιότητα (string similarity).

➢ Spatio Textual Boolean OR Query

Δοθέντος ενός χωροκειμενικού αντικειμένου t που περιέχει χωρική(t.lat, t.lon) και κειμενική(t.text) πληροφορία καθώς και μία ακτίνα r, ένα Boolean AND Query επιστρέφει όλα τα χωροκειμενικά query αντικείμενα(Q) τα οποία βρίσκονται εντός της ακτίνας r από το γεωγραφικό στίγμα που εκφράζεται από τις συντεταγμένες {t.lat, t.lon}, δηλαδή dist({q.lat, q.lon},{t.lat, t.lon})<=r, και ικανοποιεί ένα ποσοστό της κειμενική πληροφορίας t.text, δηλαδή t.text ∈ q.text. Το μέρος της κειμενικής πληροφορίας το οποίο πρέπει να ικανοποιείται για να επικυρωθεί μια ζεύξη αποτελεί όρισμα του αλγορίθμου. Αν υποθέταμε ότι στο παράδειγμα μας (βλ. Εικόνα 1), το σημείο ελάχιστης κειμενικής ομοιότητας (similarity threshold) ήταν το query να περιέχει τουλάχιστον μία λέξη κοινή με εκείνες του t.text, τότε τα αντικείμενα που επαληθεύουν το ερώτημα θα ήταν με το q1, το q2 και το q3.

➢ Spatio Textual kNN Query

Δοθέντος ενός χωροκειμενικού αντικειμένου t που περιέχει χωρική {t.lat, t.lon} και κειμενική(t.text) πληροφορία μία ακτίνα r καθώς και ενός k το οποίο ορίζει το πλήθος των αποτελεσμάτων, ένα kNN Query επιστρέφει τα k χωροκειμενικά query αντικείμενα (Q) τα οποία βρίσκονται εντός της ακτίνας r από το γεωγραφικό στίγμα που εκφράζεται από το {t.lat, t.lon}, δηλαδή dist({q.lat, q.lon},{t.lat, t.lon})<=r, αλλά και πιο κοντά στο t από τα υπόλοιπα, ενώ παράλληλα παρουσιάζουν κειμενική ομοιότητα. Θέλουμε να επιστρέψουμε τα k ορισμένα γεωγραφικά πλησιέστερα αντικείμενα $q \in Q$ τα οποία παρουσιάζουν μερική κειμενική ομοιότητα. Έτσι, αν για τα δεδομένα της Εικόνας 1 είχε δοθεί k=2 τότε τα αντικείμενα που θα επιστρέφονταν θα ήταν το q1 και q2 καθώς το q4 αν και εντός r και με μερική κειμενική ομοιότητα δεν είναι στα πρώτα 2 πλησιέστερα στο t query αντικείμενα.

➢ Spatio Textual Top-k Query

Δοθέντος ενός χωροκειμενικού αντικειμένου t που περιέχει χωρική {t.lat, t.lon} και κειμενική(t.text) πληροφορία, ενός συνόλου Q χωροκειμενικών ερωτημάτων καθώς και ενός k, ένα Top-k Query χρησιμοποιεί μια ranking συνάρτηση η οποία συνυπολογίζεται βασιζόμενη τόσο στην χωρική απόσταση όσο και στην κειμενική ομοιότητα για να επιστρέψει τα k αντικείμενα με την υψηλότερη βαθμολογία (ranking) ως προς το δοθέν χωροκειμενικό αντικείμενο t. Η συνάρτηση για τον υπολογισμό ενός spatio-textual ranking είναι:

$$STRanking(t, q) = \alpha \cdot Dist(\{q.lat,\ q.lon\}, \{t.lat,\ t.lon\}) + (1 - \alpha) \cdot TextSim(t.text,\ q.text)$$

Όπου Dist({q.lat, q.lon},{t.lat, t.lon}) η επιλεγμένη συνάρτηση για τον υπολογισμό της γεωγραφικής απόστασης μεταξύ του t και q, $TextSim(t.text,\ q.text)$ η επιλεγμένη συνάρτηση για τον υπολογισμό της κειμενικής ομοιότητας μεταξύ του t και q και $\alpha \in [0,1]$ μια σταθερά για την εναλλαγή της δοσμένης βαρύτητας μεταξύ της χωρικής και κειμενικής ομοιότητας.

➢ Spatio Textual Similarity Join

Δοθέντων δύο συνόλων χωροκειμενικών αντικειμένων T και Q, ένα Spatio Textual Similarilarity Join επιστρέφει τα ζεύγη εκείνα που βρίσκονται εντός μίας δοσμένης απόστασης r, $Dist\{q.lat,\ q.lon\}, \{t.lat,\ t.lon\} < r$, και δοθέντος ενός κατωφλιού κειμενικής ομοιότητας (similarity threshold) α, επαληθεύεται μία συνάρτηση εύρεσης κειμενικής ομοιότητας $TextSim(q.text, t.text) \geq a$.

2.2 Τεχνικές και Τεχνολογίες Επεξεργασίας Μεγάλων Δεδομένων

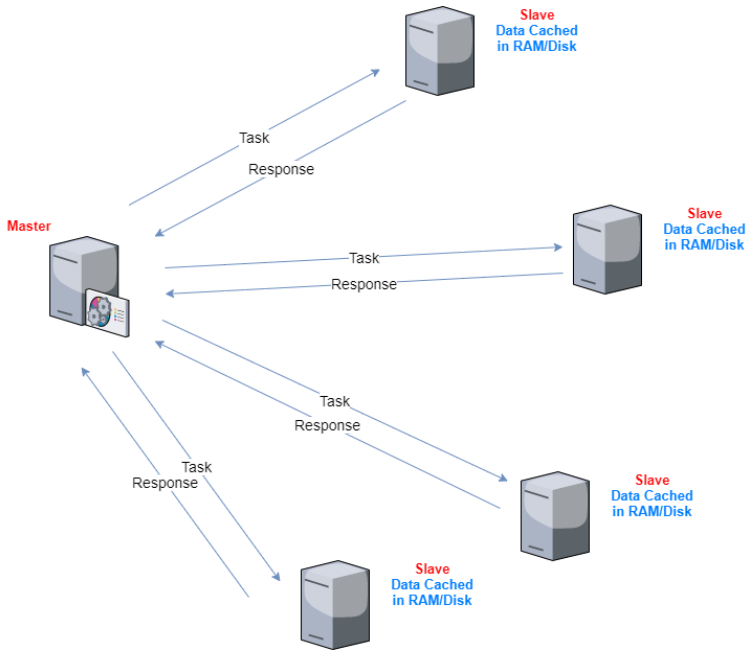
Κεντρικοποιημένα συστήματα

“Κεντρικοποιημένο” ονομάζεται ένα σύστημα το οποίο εκτελεί τις επεξεργασίες του σε έναν και μόνο ανεξάρτητο υπολογιστή. Η διαχείριση των δεδομένων, η επεξεργασία και ο υπολογισμός των αποτελεσμάτων προέρχονται αποκλειστικά και μόνο από την διαχείριση και χρήση των πόρων αυτού του υπολογιστή στο σύστημα. Ένα από τα πλεονεκτήματα ενός κεντρικοποιημένου συστήματος είναι πως δεν απαιτείται κάποια μεταφορά των δεδομένων μέσω δικτύου, καθώς έχουμε την παρουσία ενός και μόνο κόμβου επεξεργασίας, κάτι που αποτρέπει την κατανάλωση δικτυακών πόρων και καθυστέρηση.

Κατανεμημένα Συστήματα

Ο μεγάλος αριθμός δεδομένων και οι ανάγκες που προέκυψαν με σκοπό την επεξεργασία αυτών οδήγησαν στην δημιουργία των τεχνολογιών Μεγάλων Δεδομένων. Τις ανάγκες αυτές καλούνται να επιλύσουν τα κατανεμημένα συστήματα κάνοντας χρήση τεχνικών παράλληλης επεξεργασίας και διαμοιρασμού του φόρτου των εργασιών. Ένα κατανεμημένο σύστημα επεξεργασίας κάνει διαχείριση των πόρων, της επεξεργαστικής ισχύος και της αποθήκευσης των δεδομένων με στόχο τον ισομερή διαμοιρασμό των υπολογιστικών διαδικασιών. Η αρχιτεκτονική ενός τέτοιου συστήματος μπορεί να είναι η διαχείριση των πυρήνων ενός επεξεργαστή σε έναν και μόνο υπολογιστή (standalone system) αλλά και η ενορχήστρωση (orchestration) ενός ολοκληρωμένου συμπλέγματος (cluster) υπολογιστών (βλ. Εικόνα 2).

Γίνεται αντιληπτό πως η ανάγκη για αξιόπιστες και αποτελεσματικές υλοποιήσεις κατανεμημένων εργαλείων ήταν αυξημένη, γεγονός που οδήγησε διαφορετικές προσεγγίσεις επίλυσης. Στη συνέχεια παραθέτουμε μερικές εξ αυτών.



***Εικόνα 2** - Αρχιτεκτονική ενος κατανεμημένου συμπλέγματος (cluster) υπολογιστών*

Batch Processing

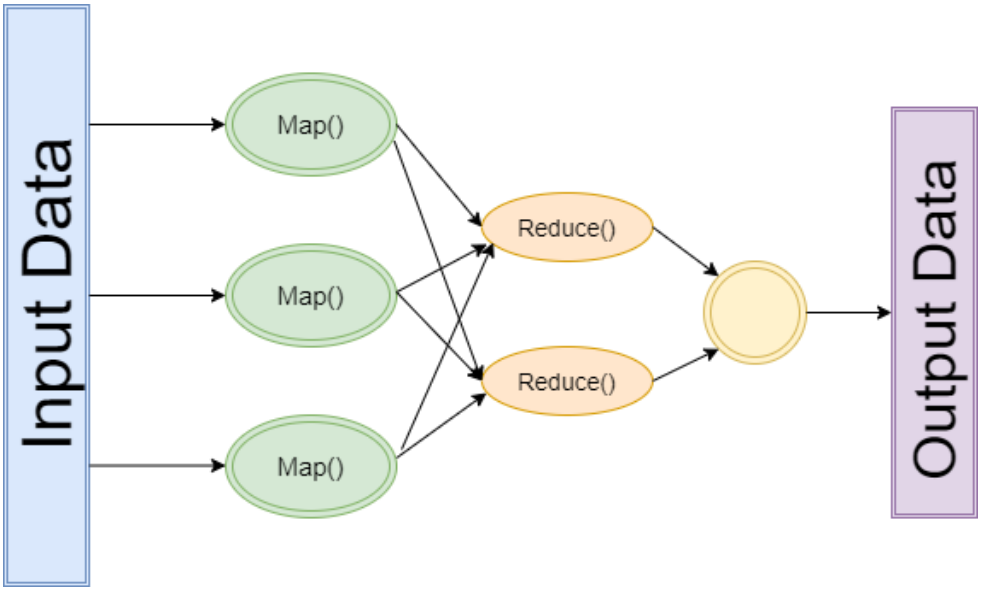
Με το όρο batch processing στα κατανεμημένα συστήματα αναφερόμαστε στην επεξεργασία ενός καθορισμένου συνόλου δεδομένων η οποία επιτυγχάνεται μέσα από τον ορισμό συγκεκριμένων εργασιών (jobs). Κατά το batch processing, έπειτα από την συλλογή και αποθήκευση των δεδομένων, επέρχεται η επεξεργασία αυτών. Το batch processing βελτιώνει την αποτελεσματικότητα μέσω του καθορισμού προτεραιοτήτων επεξεργασίας και της εκτέλεσης και ολοκλήρωσης συγκεκριμένων εργασιών (jobs) σε στρατηγικά καθορισμένες στιγμές, ώστε να ευνοηθεί το σύνολο των εργασιών και να επιτευχθεί το βέλτιστο αποτέλεσμα.

MapReduce

Η ανάγκη για αποδοτική επεξεργασία Μεγάλων Δεδομένων (Big Data) ώθησε την επιστημονική κοινότητα στην αναζήτηση νέων τεχνικών και εργαλείων.
Το MapReduce[13] είναι ένα προγραμματιστικό μοντέλο το οποίο επιτυγχάνει αποτελεσματική και γρήγορη επεξεργασία μεγάλου όγκου δεδομένων, αξιοποιώντας την παράλληλη επεξεργασία σε ένα cluster υπολογιστών. Το μοντέλο του MapReduce (βλ. Εικόνα 3) χωρίζεται σε δύο διακριτές φάσεις:

- **Map:** Ένα σύνολο ανεξάρτητων κομματιών των συνολικών δεδομένων ανατίθεται σε διαφορετικές διεργασίες (mappers) οι οποίες παράγουν ως αποτέλεσμα ένα set από <key, value> pairs.

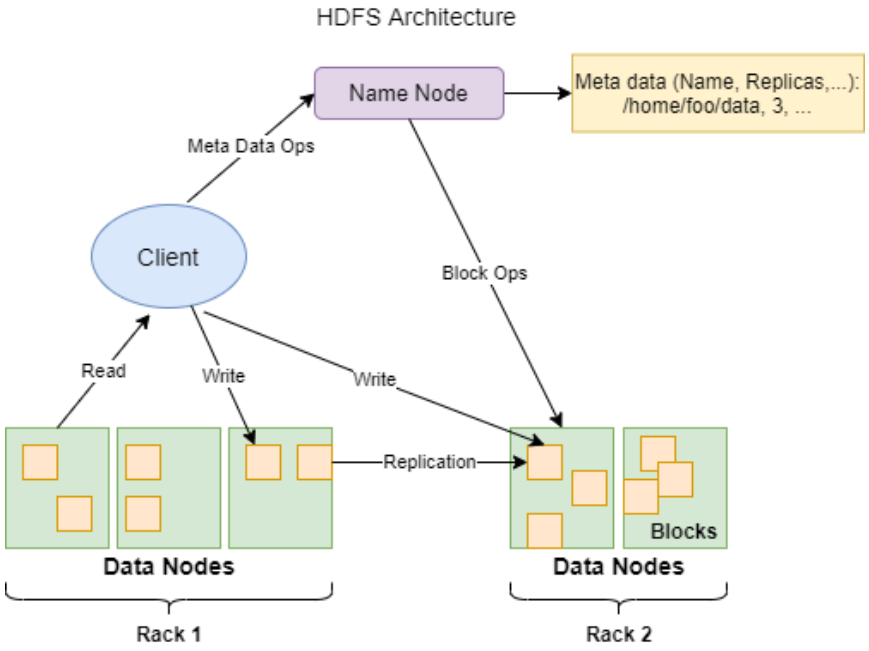
- **Reduce:** Τα <key, value> από τη φάση Map διοχετεύονται σε ένα σύνολο από διεργασίες (reduce) οι οποίες χρησιμοποιώντας custom functions παράγουν τα συνολικά και τελικά <key, value> pairs των αποτελεσμάτων.



***Εικόνα 3** - Το μοντέλο MapReduce*

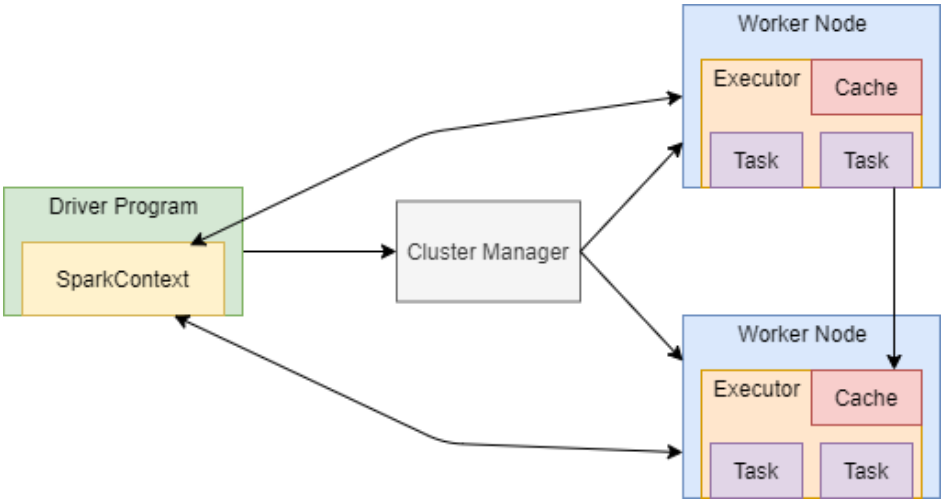
Γνωστές εφαρμογές του MapReduce και εργαλεία που αξιοποιούν το κατανεμημένο μοντέλο εργασιών του είναι:

- **Apache Hadoop:** Ανοιχτού κώδικα υλοποίηση του προγραμματιστικού μοντέλου MapReduce.
- **HDFS:** Ένα κατανεμημένο σύστημα αρχείων, που παρέχει υψηλό βαθμό εξυπηρέτησης από κάθε μονάδα του cluster σε ένα μεγάλο σύνολο δεδομένων.



Εικόνα 4 *- Αρχιτεκτονική HDFS και παράδειγμα κατανομής πακέτων πληροφορίας*

- **ZooKeeper:** Μια κεντρική, υψηλής απόδοσης υπηρεσία για κατανεμημένες εφαρμογές, με σκοπό τη διατήρηση πληροφοριών διαμόρφωσης, ονομασιών, παροχής κατανεμημένου συγχρονισμού και παροχής ομάδων.
- **Apache Spark:** Μία ανοιχτού κώδικα υλοποίηση του MapReduce μοντέλου που έχει σχεδιαστεί για να ενισχύει την υπολογιστική ταχύτητα. Το Hadoop MapReduce προσφέρει ανάγνωση και εγγραφή από τον δίσκο με αποτέλεσμα να επιβραδύνει τον υπολογισμό. Αντίθετα, το Apache Spark μπορεί να τρέξει στην κορυφή του Hadoop, χρησιμοποιώντας την RAM memory για ανάγνωση και εγγραφή, παρέχοντας μια αποδοτικότερη υπολογιστική λύση.

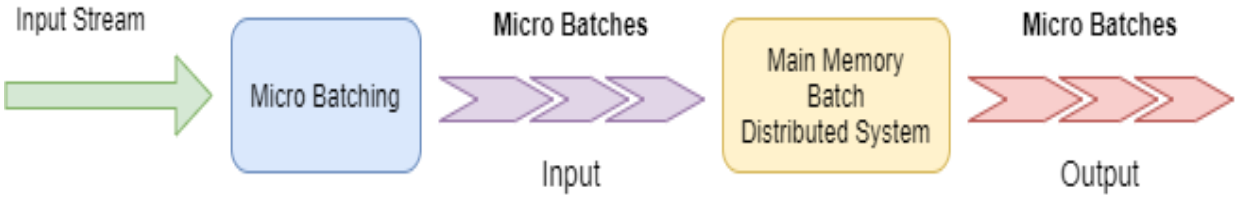


Εικόνα 5 *- Τοπολογία ενός Apache Spark συστήματος*

Stream Processing

Η παρούσα έρευνα επικεντρώθηκε στον κλάδο της επιστήμης των Μεγάλων Δεδομένων, γνωστό ως Stream Processing. Το Stream Processing είναι μία τεχνολογία των Big Data η οποία χρησιμοποιείται για αναζήτηση και επεξεργασία συνεχόμενων ροών δεδομένων γρήγορα και εντός μιας μικρής χρονικής περιόδου από την λήψη των δεδομένων. Η χρονική αυτή περίοδος μπορεί να κυμαίνεται από μερικά χιλιοστά του δευτερολέπτου έως και λίγα λεπτά. Η αξία των τεχνικών επεξεργασίας συνεχόμενων ροών δεδομένων γίνεται αντιληπτή αν αναλογιστούμε τρόπους επεξεργασίας αυτών με τεχνικές και τεχνολογίες batch processing. Σε μια Batch προσέγγιση, όπως αυτή του Hadoop ecosystem, θα έπρεπε να λάβουμε ένα κομμάτι της ροής, στην συνέχεια να τη διακόψουμε, να αποθηκεύσουμε το batch στο κατανεμημένο file system (HDFS), να προβούμε σε όλες τις απαραίτητες διεργασίες orchestration και κατανομής εργασιών και τελικά, στην εκτέλεση αυτών. Έπειτα, θα έπρεπε να ανοίξουμε ξανά τη ροή ώστε να λάβουμε την επόμενη παρτίδα (batch), μην ελέγχοντας τον αριθμό των εγγραφών ή batch και επαναλαμβάνοντας αυτόν τον εξαντλητικό σε πόρους κύκλο εργασιών (pipeline). Αντίθετα, οι τεχνολογίες Streaming processing όπως το Apache Spark Streaming και το Apache Storm κάνουν δυνατή την Real-Time επεξεργασία γρήγορα, αποδοτικά, προσφέροντας fault tolerance και αξιοποιώντας τους διαθέσιμους, κατανεμημένους πόρους του συστήματος βέλτιστα.

Οι τεχνολογίες Streaming Processing χωρίζονται σε δύο βασικές κατηγορίες οι οποίες διακρίνονται από τον χρόνο απόκρισης και επεξεργασίας των εισερχόμενων ροών. Έτσι, εργαλεία όπως το Apache Storm χρησιμοποιούν αρχιτεκτονική με στόχο την Real-Time επεξεργασία των δεδομένων, εξαλείφοντας την καθυστερημένη απόκριση (latency), ενώ άλλα, όπως το Apache Spark Streaming, χωρίζουν τις ροές σε micro-batches ορίζοντας ένα χρόνο απόκρισης μεταξύ αυτών.



Εικόνα 6 *- Επεξεργασία συνεχόμενης ροής micro-batches*

2.3 Τεχνικές Επεξεργασίας Συνεχόμενων Χωροκειμενικών Ροών

Όπως αναφέρθηκε και στο τέλος της προηγούμενης ενότητας, υπάρχουν αρκετά εργαλεία επεξεργασίας συνεχόμενων ροών δεδομένων, όπως το Apache Spark Streaming, το Apache Storm και το Apache Flink. Παρατηρείται, όμως, έλλειψη ενός framework εξιδανικευμένου για την επεξεργασία και εξυπηρέτηση των αναγκών των χωροκειμενικών δεδομένων που θα τροφοδοτούνται σε αυτό μέσω της συνεχόμενης ροής.

Πρωταρχικός στόχος αυτής της έρευνας είναι ο σχεδιασμός και η ανάπτυξη ενός εργαλείου επεξεργασίας Streaming Data προσφέροντας την εξειδίκευση και παραμετροποίηση αυτού ώστε να εξυπηρετηθούν οι ανάγκες για την αποτελεσματική και όσο το δυνατόν βέλτιστη επεξεργασία, ζεύξη χωροκειμενικών δεδομένων και ερωτημάτων.

Related Work Τεχνικών και Μεθόδων

Το πρώτο μέρος της έρευνάς μας επικεντρώθηκε στην μελέτη εργαλείων που επεκτείνουν τις δυνατότητες ενός συστήματος διαχείρισης συνεχόμενων ροών δεδομένων, προσθέτοντάς του δυνατότητες διαχείρισης χωροκειμενικών ροών.
Συγκεκριμένα, μελετήθηκε ο τρόπος λειτουργίας πέντε εργαλείων:

- Tornado[1, 2]
- Skype[3]
- DSkype[3]
- PS2Stream[4]
- SSTD[7]

Όλα τα παραπάνω, εξαιρουμένου το Skype που έχει κεντρικοποιημένη (centralized) αρχιτεκτονική, υλοποιούνται επεκτείνοντας τις δυνατότητες επεξεργασίας συνεχόμενων ροών του Apache Storm, το οποίο είναι ένα από τα πλέον διαδεδομένα συστήματα ροών δεδομένων. Με τον τρόπο χρησιμοποιούν την αρχιτεκτονική του Apache Storm το οποίο όμως δεν υποστηρίζει αποτελεσματικούς μηχανισμούς διαχείρισης χωροκειμενικών αντικειμένων και ερωτημάτων (queries).
Στη συνέχεια, θα αναλύσουμε τα τεχνικά χαρακτηριστικά και την τοπολογία που διακρίνουν το καθένα και θα ασχοληθούμε με τα distributed συστήματα μη σχολιάζοντας περαιτέρω το Skype λόγω της κεντρικοποιημένης του τοπολογίας. Η distributed έκδοσή του, DSkype, μετρήθηκε 26~49% ταχύτερη από την κεντρικοποιημένη έκδοση. Το γεγονός αυτό μας δίνει μια καλή εικόνα για την υπεροχή και τα πλεονεκτήματα ενός ορθά κατανεμημένου συστήματος συγκριτικά με μία κεντρικοποιημένη λύση σχετικά με την απάντηση χωροκειμενικών ερωτημάτων συνεχόμενων ροών δεδομένων.

	SpatioTextual Queries	Topology	Storing/Distribution/Routing
Tornado	-Spatial range Boolean AND -Spatial range Boolean OR -kNN	Two Layer Indexing -Routing Layer -Evaluation Layer	-A-Grid (non-overlapping rectangular Partitioning) -FAST (spatial-keyword index: Spatial-pyramid, Hybrid of inverted lists and keyword frequencies)
DSkype	Top-k Over sliding window	-Subscription/Message bolts -Distribution bolts -Subscription bolts -Message bolts -Aggregation bolts	-Hashing Based Method -Location-based Method -Keyword-based Method -Prefix-based Method
PS2Stream	-Spatial range Boolean AND -Spatial range Boolean OR	-Dispatcher -Worker -Merger	-Hybrid partitioning algorithm *Βασιζόμενος ανάλογα με την ομοιότητα των στοιχείων και queries είτε χωρικά είτε κειμενικά με σκοπό το workload balancing* -Grid based partitioning *Κάθε partition περιέχει ένα hash map τόσο για τα χωρικά όσο και κειμενικά στοιχεία που περιέχει*
SSTD	-Spatial range Boolean AND -kNN -Top-k frequent-term	-Routers -Workers	- QT (Quad-Text) tree

Πίνακας 1 *- Συστήματα Διαχείρισης Χωροκειμενικών Ροών*

Σχόλια και παρατηρήσεις

Το Tornado μοιάζει να έχει ένα πολύ ενδιαφέρον μηχανισμό partitioning καθώς και adaptive workload balancing μη έχοντας downtime. Στο experimental evaluation του Tornado η ομάδα υλοποίησης έδειξε πως το Fast indexing αποδίδει καλύτερο από το Grid-Inverted-Index (GI2) που χρησιμοποιεί το PS2Stream. Αυτό αποδίδεται στο memory overhead εξαιτίας των duplications των queries σε όλο το spatial-grid. Επίσης εξετάστηκε η απόδοση του A-Grid σε σχέση με το κλασσικό Grid αλλά και R-tree. Αποδείχτηκε πως ενώ το A-Grid και R-tree αποδίδουν εξίσου καλά στο point routing, το A-Grid έχει καλύτερη απόδοση σε range routing. Το DSkype είναι το μοναδικό εκ των τριών το οποίο απαντάει σε top-k ερωτήματα, κάνοντας χρήση ενός Cost-based

K-Skyband. Το DSkype κάνει χρήση πολλών cost-function ώστε να εντοπίσει την ομοιότητα αλλά και να διατηρήσει το workload balance.
Επίσης οι μετρικές του DSkype έδειξαν πως μέχρι ένα πλήθος εγγραφών προς επεξεργασία η prefix-based μεθοδος αποδίδει καλύτερα ενώ από ένα πλήθος εγγραφών και μετά εξαιτίας του μεγάλου αριθμού duplications η hash-based μεθοδος αποδίδει καλύτερα. Bottleneck της τελευταίας είναι το communication cost μεταξύ των partitions το οποίο όμως είναι μικρότερο σε σχέση με αυτό των duplications από ένα πλήθος και έπειτα.

Όπως αναφέρθηκε, όλα τα παραπάνω συστήματα είναι σχεδιασμένα για tuple-at-time εργαλεία, όπως το Apache Storm, και όχι για micro-batching, όπως το Apache Spark Streaming. Μία ακόμα κοινή διαπίστωση είναι η ανάγκη για αποτελεσματικό καταμερισμό των δεδομένων σε partitions κατανεμημένα στους workers του cluster για την παράλληλη επεξεργασία αυτών, κάτι που στην πλειονότητα υλοποιήθηκε με κάποιο Spatial-first indexing σε Spatial Grid.

Spatial-first και Text-First Indexing

Κατά το Spatial-first approach τα χωροκειμενικά αντικείμενα κατανέμονται με βάση την spatial πληροφορία τους στα Cells του Grid και η συσχετίσεις μεταξύ τους γίνονται εντός αυτών βάσει της text πληροφορίας που φέρουν. Αντίθετα κατά το Text-First Approach χρησιμοποιώντας μία συνάρτηση κειμενικής ομοιότητας τα χωροκειμενικά αντικείμενα χωρίζονται στα Cells και έπειτα πραγματοποιούνται χωρικές συγκρίσεις απόστασης μεταξύ των σημείων με σκοπό την επίτευξη των τελικών ζεύξεων.

Τα συμπεράσματα από όλα τα προαναφερθέντα συστήματα επεξεργασίας χωροκειμενικών ροών δείχνουν να συμφωνούν πως το spatial-first approach υπερισχύει της text-first κατά την οποία το partitioning των αντικειμένων καθορίζεται από inverted files ανά Cell που περιέχουν το σύνολο της κειμενικής πληροφορίας, για παράδειγμα, μια λίστα λέξεων που περιλαμβάνονται στα δεδομένα που τους έχουν ανατεθεί. Ένα ακόμα κομβικό χαρακτηριστικό που φαίνεται να μοιράζονται όλα τα εργαλεία είναι κάποιος μηχανισμός adaptive partitioning. Κάτι τέτοιο επιτυγχάνεται χρησιμοποιώντας μια analytics ή rating function, η οποία είναι υπεύθυνη για τον υπολογισμό, την πρόβλεψη και τελικά την εξαγωγή του βαθμού κατανομής των δεδομένων. Βασικές μετρικές αποτελεσματικότητας αποτελούν ο ισομοιρασμός των αντικειμένων στα Cells, αλλά και το αναγκαίο πλήθος των αντιγράφων (Duplications) που μπορεί να προκύψουν.

3. Ορισμός του προβλήματος

3.1 Παρουσίαση και βασική λογική επίλυσης

Όπως αναφέρθηκε στα προηγούμενα κεφάλαια το σύστημα το οποίο καλούμαστε να αναπτύξουμε ακολουθεί την τοπολογία ενός Publish and Subscribe μοντέλου στο οποίο θα υλοποιήσουμε Spatiotextual Similarity Joins Συνεχόμενων Ροών δεδομένων και ενός Batch συνόλου. Ένα τέτοιο σύστημα βρίσκει εφαρμογή σε περιπτώσεις E-Coupon promotions, σε ανάλυση των ροών δεδομένων που αποστέλλονται από κινούμενα αντικείμενα ανά την υφήλιο (π.χ. αεροπλοΐα και ναυσιπλοΐα), αλλά και σε οποιαδήποτε εφαρμογή ή σύστημα επεξεργασίας χωροκειμενικού περιεχομένου σε πραγματικό χρόνο.

Στα πλαίσια της παρούσας έρευνας επιλέχθηκε η ανάπτυξη ενός συστήματος βασισμένου σε επεξεργασία ροών χωριζόμενων σε micro-batches, όπως φαίνεται στην Εικόνα 6.

Ένα απλός τρόπος επίλυσης του παραπάνω προβλήματος θα ήταν η ανάπτυξη ενός συστήματος το οποίο, δοθέντος ενός συνόλου, πλήθους Β στατικών δεδομένων (Batch Data), τα οποία περιέχουν χωρική και κειμενική πληροφορία, να τροφοδοτείται ανά τακτά χρονικά διαστήματα κάποιων δευτερολέπτων με μικρο-δέσμες (micro-batch) αντικειμένων μέσω μιας συνεχόμενης ροής χωροκειμενικών δεδομένων, πλήθους S, μέσω μιας συνεχόμενης ροής. Στη συνέχεια, το σύστημα γνωρίζοντας την μέγιστη δυνατή απόσταση (range) και μία ελάχιστη τιμή η οποία θα εκφράζει την κειμενική ομοιότητα, θα πραγματοποιούσε BxS συγκρίσεις με στόχο τη ζεύξη (similarity joins) αντικειμένων των δύο συνόλων.

3.2 Ανάγκες και προκλήσεις

Η παραπάνω αποτελεί μία απλουστευμένη στρατηγική επίλυσης. Αντίθετα, για μία ορθή και αποδοτική επίλυση θα πρέπει να λάβουμε υπόψη κάποια από τα προβλήματα που προκύπτουν.

Αρχικά, λόγω του μεγέθους και της φύσης των δεδομένων, καθώς και του πλήθους των υπολογισμών, η επεξεργασία των αντικειμένων και η πραγματοποίηση ζεύξεων, μας οδηγούν σε δύο προβληματισμούς. Πρώτον, με το ερώτημα της υπολογιστικής ισχύος που θα χρειαστεί το σύστημα μας, και δεύτερον, με την αρχιτεκτονική σχεδίαση αυτού.

Καταλαβαίνουμε εύκολα πως πρόκειται για ένα πρόβλημα με μεγάλο μέγεθος στατικών δεδομένων, τα οποία θα συζευγνύονται με πολλαπλές μικρο-δέσμες μέσω πολλαπλών μετασχηματισμών και υπολογισμών χωρικής και κειμενικής φύσης. Για την αντιμετώπιση των αναγκών αυτών, λοιπόν, θα πρέπει να αντιμετωπίσουμε το παραπάνω πρόβλημα σχεδιάζοντας το σύστημά μας με μία κατανεμημένη αρχιτεκτονική και με τέτοιο τρόπο ώστε να είναι ικανό για ταχύτατες επεξεργασίες που θα το καταστήσουν κατάλληλο για αντιμετώπιση υπολογισμών και πραγματοποίηση similarity joins με αντικείμενα συνεχόμενων ροών.

Επιπλέον σημεία για σκέψη αποτελούν ο διαμοιρασμός των δεδομένων σε ένα κατανεμημένο σύστημα (Distributed System), ο αριθμός των συζεύξεων καθώς και το πλήθος των μεταφορών στοιχείων μεταξύ των nodes του cluster του συστήματος. Για να γίνει εύκολα κατανοητό αυτό μπορούμε να σκεφτούμε ένα παράδειγμα όπου θα θέλαμε να επιτύχουμε ζεύξη μεταξύ χωροκειμενικών αντικειμένων μόνο αν η χωρική πληροφορία που φέρουν τα τοποθετεί στην ίδια ήπειρο, ενώ ταυτόχρονα μοιράζονται τουλάχιστον μία κοινή λέξη στο κειμενικό τους περιεχόμενο. Δεν θα υπήρχε λόγος να αντιμετωπίσουμε το παραπάνω πρόβλημα ελέγχοντας όλα τα στατικά αντικείμενα με όλα τα αντικείμενα κάθε micro-batch· κάτι τέτοιο θα οδηγούσε σε πολυπλοκότητα O(NxN). Αντιθέτως, θα μπορούσαμε να κατανέμουμε τα αντικείμενα των παραπάνω συνόλων ανά ήπειρο, κάνοντας χρήση της χωρικής ταυτότητας τους και στη συνέχεια να διεξάγουμε έλεγχο χωρικής ομοιότητας μεταξύ των αντικειμένων αυτών. Αυτό θα μείωνε αρκετά την πολυπλοκότητα του αλγορίθμου μας.

3.3 Σχεδιασμός και αντιμετώπιση προβλήματος

Λαμβάνοντας υπόψη τις παραπάνω ανάγκες, για την ανάπτυξη του συστήματός μας κάνουμε χρήση μιας κατανεμημένης αρχιτεκτονικής. Χρησιμοποιώντας το εργαλείο Apache Spark Streaming, η αρχιτεκτονική αυτή είναι κατάλληλη για επεξεργασία συνεχόμενων ροών micro-batch δεδομένων. Παράλληλα, κατανέμουμε στρατηγικά τα δεδομένα σε εικονικά διαμερίσματα (partitions) του cluster (Partitioning) με στόχο την ουσιώδη μείωση των υπολογισμών. Χρησιμοποιούμε τεχνικές αντιγραφής μεταξύ τους (Duplication), βάσει της χωρικής ταυτότητας (Spatial-First), ώστε να ελαχιστοποιήσουμε την μεταφορά πληροφορίας μεταξύ των κόμβων του συστήματος (Cluster Nodes). Τέλος, κάνοντας κατάλληλη διαχείριση και μετασχηματισμό των εισερχόμενων micro-batch δεδομένων, εφαρμόζουμε τεχνικές MapReduce και εντέλει επιτυγχάνουμε την αποτελεσματική και αποδοτική ζεύξη των αντικειμένων.

Ορίζουμε ένα στατικό σύνολο χωροκειμενικών δεδομένων Q (Static Data). Επίσης, θεωρούμε ένα σύνολο T το οποίο αντιπροσωπεύει τα χωροκειμενικά δεδομένα που

εισέρχονται στο σύστημα μας μέσω μιας ορισμένης συνεχόμενης ροής (Streaming Data). Θα διεξάγουμε Spatio Textual Similarity Join μεταξύ των συνόλων αυτών, με στόχο τη ζεύξη των αντικειμένων που περιέχονται στα δύο σύνολα, και πληρούν τόσο τους χωρικούς όσο και τους κειμενικούς περιορισμούς των συνθηκών ζεύξης.

Spatial Join using Haversine formula

Για τον ορθό υπολογισμό της απόστασης μεταξύ των δεδομένων μας, θα χρησιμοποιηθεί ο τύπος της απόστασης Haversine[4], ο οποίος υπολογίζει την απόσταση μεταξύ δύο σημείων τοποθετημένων στη επιφάνεια μιας σφαίρας, βασιζόμενος στις γεωγραφικές συντεταγμένες των σημείων αυτών (βλ. Εικόνα 7).

$$\Delta\sigma = 2\,\mathrm{r} * \sin^{-1}\left(\sqrt{sin^2\left(\frac{\Delta\Phi}{2}\right) + \cos\varphi_s \cos\varphi_f \, sin^2\left(\frac{\Delta\lambda}{2}\right)}\right)$$

Εικόνα 7

Δοθείσης μιας απόστασης r, η χωρική ζεύξη μεταξύ ενός χωρικού αντικειμένου q ∈ Q και ενός t ∈ T επιτυγχάνεται όταν:

$$HavDist(\{q_{lat}, q_{lon}\}, \{t_{lat}, t_{lon}\}) \leq r$$

Όπου $HavDist(\{q_{lat}, q_{lon}\}, \{t_{lat}, t_{lon}\})$ η απόσταση μεταξύ των συντεταγμένων των αντικειμένων t και q, και r η ορισμένη απόσταση.

Textual Join using Jaccard Similarity

Η κειμενική ζεύξη δύο αντικειμένων που φέρουν κειμενική πληροφορία γίνεται με χρήση της συνάρτησης κειμενικής ομοιότητας Jaccard. Η συνάρτηση Jaccard υπολογίζει το βαθμό ομοιότητας (Similarity Rating) για δύο λίστες λέξεων, t_{set1},t_{set2}, ως τον λόγο των κοινών λέξεων των συνόλων $t_{set1} \cap t_{set2}$ προς το σύνολο όλων των λέξεων αυτών, $t_{set1} \cup t_{set2}$ (βλ. Εικόνα 8).

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

Εικόνα 8

Δοθέντων δύο αντικειμένων $q \in Q$ και $t \in T$, καθώς και ενός threshold a, η κειμενική ζεύξη μεταξύ τους θα πραγματοποιηθεί όταν ισχύει:

$$J(q_{text}, t_{text}) \geq a$$

3.4 Στόχος Συστήματος

Ο στόχος μας είναι ο σχεδιασμός και η ανάπτυξη ενός συστήματος που θα υλοποιεί ένα Publish/Subscribe μοντέλο πραγματοποιώντας ερωτήματα Spatio Textual Similarity Join μεταξύ ενός Batch συνόλου Q χωροκειμενικών αντικειμένων και μιας συνεχόμενης ροής χωροκειμενικών δεδομένων T. Πρωταρχικός σκοπός μας είναι η παρουσίαση μιας αποτελεσματικής και αποδοτικής λύσης βασιζόμενοι στην κατανομή των δεδομένων και στην παράλληλη επεξεργασία των διεργασιών επεξεργασίας και ζεύξης. Κάτι τέτοιο θα επιτευχθεί χρησιμοποιώντας πλέον αποτελεσματικούς μηχανισμούς ευρετηρίασης, διαμοιρασμού (Partitioning) και μειώνοντας στο βέλτιστο την περιττή μεταφορά πληροφορίας μεταξύ των κόμβων του συστήματος.

4. Μεθοδολογία και Ανάλυση Συστήματος

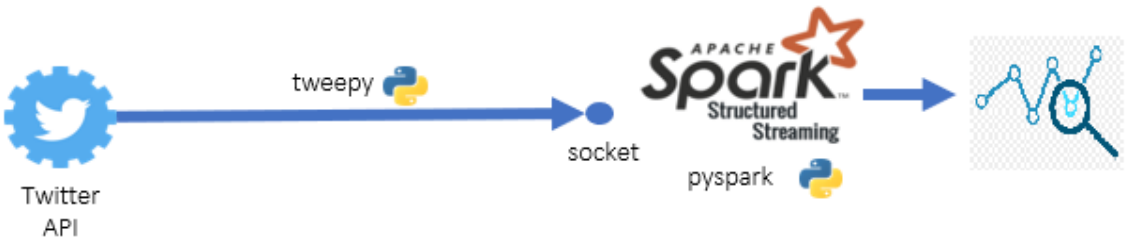
4.1 Παρουσίαση συστήματος

Για την υλοποίηση του συστήματος, θεωρούμε ένα σύνολο Q από batch χωροκειμενικά αντικείμενα στα οποία θα αναφερόμαστε ως χωροκειμενικά αντικείμενα ερωτημάτων (Geospatial Query Objects). Ορίζουμε ένα σύνολο Q και κάθε χωροκειμενικό αντικείμενο που περιέχει ως $q \in Q$, όπου $q = (q_{id}, q_{lat}, q_{lon}, q_{text})$. Παράλληλα, θεωρούμε μια συνεχόμενη ροή χωροκειμενικών αντικειμένων $t \in T$. Στα πλαίσια των αναγκών του συστήματος και της διεξαγωγής της πειραματικής μελέτης χρησιμοποιήθηκε μία συνεχόμενη ροή πραγματικού χρόνου (Real-Time) tweet αντικειμένων (object). Ορίζουμε tweet object κάθε αντικείμενο $t \in T$, όπου $t = (t_{author}, t_{lat}, t_{lon}, t_{text})$. Για την υλοποιήση του μηχανισμού αυτού έγινε χρήση του Twitter Live Streaming API και της βιβλιοθήκης Tweepy, στοχεύοντας στην ρεαλιστική εξαγωγή συμπερασμάτων της αποδοτικότητας της εφαρμογής. Κατά την εκτέλεση, ζητείται από τον χρήστη να εισάγει την απόσταση αναζήτησης (Spatial Distance) r και τον βαθμό κειμενικής ομοιότητας (Textual Threshold) a που θα χρησιμοποιηθούν κατά το χωρικό και κειμενικό Similarity Join των αντικειμένων αντίστοιχα. Ένα παράδειγμα εκτέλεσης θα μπορούσε να είναι η εύρεση όλων των συγγραφέων (authors) των tweet αντικειμένων, t_{author} $\forall$ $t \in T$, και Ids των Query objects, q_{id} $\forall$ $q \in Q$, τα οποία βρίσκονται σε μία απόσταση μικρότερη του r, όπου $HavDist([q_{lon}, t_{lon}], [q_{lat}, t_{lat}]) \leq r$, ενώ ταυτόχρονα παρατηρείται μία κειμενική ομοιότητα μεταξύ αυτών, η οποία θα επαληθεύει την συνάρτηση $J(q_{text}, t_{text}) \geq a$. Για την επίτευξη της ζεύξης χρησιμοποιούμε την συνάρτηση Haversine για τη σύγκριση και εύρεση της γεωγραφικής απόστασης μεταξύ των σημείων/ερωτημάτων καθώς και τη Jaccard similarity για την εξαγωγή ενός κειμενικού βαθμού ομοιότητας(Text Score Similarity) μεταξύ δύο κειμένων.

Για την επίτευξη της αποδοτικότερης επεξεργασίας και εκτέλεσης κρίθηκε απαραίτητη η χρήση μιας κατανεμημένης αρχιτεκτονικής στοχεύοντας στην παράλληλη εκτέλεση των εργασιών και αξιοποίηση των πόρων του συστήματος. Επιλέχθηκε να χρησιμοποιηθεί spatial-first διαχωρισμός των χωροκειμενικών δεδομένων και διαμοιρασμός (Partitioning) αυτών σε ένα πλέγμα (Grid) κάνοντας χρήση εξατομικευμένων συναρτήσεων (custom functions) για τον υπολογισμό του πλήθους των Cells αυτού, καθώς και μηχανισμών duplication για την ελαχιστοποίηση της αχρείαστης μεταφοράς δεδομένων μεταξύ των κελιών (Grid Cells) και κατ' επέκταση, κόμβους (nodes) του κατανεμημένου μας cluster.

Παράλληλα, για την βελτιστοποίηση της κατανομής φόρτου (load-balancing) των εργασιών και την αποδοτική απάντηση των πραγματικού χρόνου (Real-Time) ερωτημάτων που τίθενται στα πλαίσια της παρούσας εργασίας, έγινε χρήση και υλοποίηση αλγορίθμων βασισμένων στο προγραμματιστικό μοντέλο MapReduce. Επιλέχθηκε να χρησιμοποιηθεί το δημοφιλές εργαλείο επεξεργασίας συνεχόμενων ροών Apache Spark Streaming, το οποίο επεξεργάζεται τα δεδομένα που καταφθάνουν από τις ροές σε micro-batches (βλ. Εικόνα 2.6), εκτελώντας αποδοτικά τις απαραίτητες και αυστηρά ορισμένες εργασίες (Spark Jobs) για την εξαγωγή των επιθυμητών αποτελεσμάτων. Το Apache Spark Streaming, όπως τα περισσότερα συστήματα επεξεργασίας συνεχόμενων ροών, δεν προσφέρει κάποια εξειδικευμένη λύση (Dedicated Feature) για την επεξεργασία χωροκειμενικών δεδομένων. Χρησιμοποιώντας τις τεχνικές και τα προτερήματα που φέρει, αποτελεί ιδανική επιλογή ώστε να υλοποιήσουμε το σύστημα μας βασιζόμενοι σε αυτό (on-top of it). Για την εξατομίκευση του συστήματός μας και την αποδοτικότερη επεξεργασία χωροκειμενικών αντικειμένων πραγματοποιήθηκε διαμοιρασμός σε πλέγμα (Grid Partitioning) των δεδομένων με βάση τη χωρική τους πληροφορία (spatial first) και τη μέγιστη δυνατή απόσταση χωρικής ζεύξης r. Πραγματοποιούμε σε πρώτο χρόνο αντιγραφές (Duplications) αντικειμένων, τα οποία δύναται να ανατεθούν σε περισσότερα από ένα κελιά (Partitions), εξαλείφοντας έτσι την αχρείαστη μετακίνηση αντικειμένων μεταξύ των κελιών κατά τους υπολογισμούς και την υλοποίηση του MapReduce.

Για την υλοποίηση του receiver χρησιμοποιήθηκε η βιβλιοθήκη Tweepy, η οποία μας επιτρέπει την επικοινωνία με το Twitter Live API καθώς και την ανάθεση παραμέτρων, όπως το φιλτράρισμα των tweets με γεωγραφικό περιεχόμενο (χωρική πληροφορία) ή την επιλογή της γεωγραφικής περιοχής όπου θα ανήκουν τα αντικείμενα των micro-batches που θα εισέρχονται στο Apache Spark Streaming σύστημα μας. Μια υψηλού επιπέδου (High Level) οπτικοποίηση του συστήματός μας και των εφαρμογών που το αποτελούν (Components Pipeline) αποδίδεται στην Εικόνα 9.

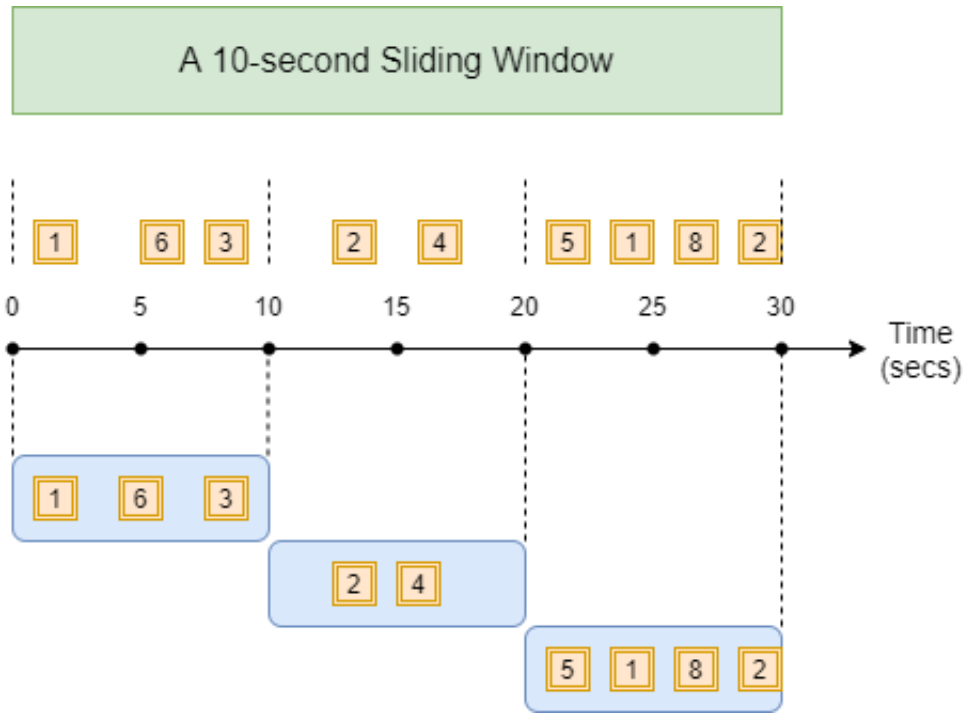


Εικόνα 9 *- Components Pipeline*

4.2 Εργαλεία και Τεχνολογίες

4.2.1 Apache Spark Streaming

Το Apache Spark Streaming επεξεργάζεται τη ροή δεδομένων σε micro-batches. Λαμβάνει μια ροή δεδομένων (Data Stream) και τη διαιρεί σε πακέτα πληροφορίας (Batches) τα οποία ονομάζονται DStreams. Στη συνέχεια πραγματοποιείται κατανεμημένη και παράλληλη επεξεργασία (Parallel Processing) των DStreams σε ένα cluster υπολογιστών. Η επεξεργασμένη πληροφορία μπορεί να αποθηκευτεί στη μνήμη (State) της εφαρμογής, ενώ παρέχεται και η δομή κυλιόμενου χρονικού παραθύρου (Sliding Window) η οποία επιτρέπει στον προγραμματιστή να αποθηκεύει αποτελέσματα για ένα συγκεκριμένο χρονικό διάστημα κάνοντας χρήση ενός κυλιόμενου χρονικά window. Στις πειραματικές διαδικασίες που πραγματοποιήθηκαν ορίσαμε τον χρόνο προσέλευσης των micro-batches στα δέκα δευτερόλεπτα. Υλοποιήθηκε ένα κυλιόμενο παράθυρο (Sliding Window) (βλ. Εικόνα 10) της τάξεως των τριών λεπτών, ενώ ορίζουμε έναν μηχανισμό ενημέρωσης του State μας.



Εικόνα 10 - Sliding Window

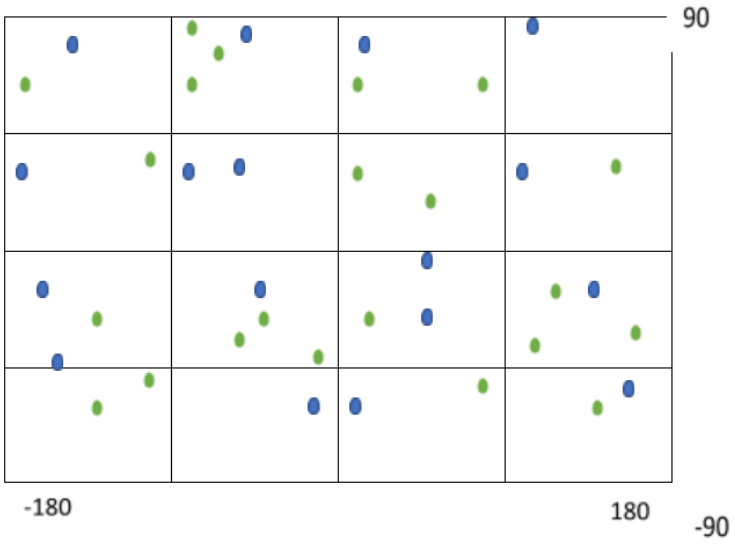
4.2.2 Twitter Stream API

Το Twitter Stream API επιστρέφει ένα JSON το οποίο αποτελείται από ένα σύνολο tweets. Κάθε tweet μπορεί να έχει πάνω από 150 χαρακτηριστικά (attributes). Στα πλαίσια των πειραματικών διαδικασιών που πραγματοποιήθηκαν έγινε χρήση των attributes που σχετίζονται με το κείμενο του tweet (κειμενική πληροφορία) καθώς και το γεωγραφικό πλάτος και μήκος (χωρική πληροφορία) όπου αναρτήθηκε το συγκεκριμένο tweet. Επίσης γίνεται χρήση των πληροφοριών σχετικά με το id και τον συγγραφέα (author) του tweet.

4.3 Partitioning

4.3.1 Δημιουργία του Uniform Grid

Όπως προαναφέρθηκε, επιλέξαμε να υιοθετήσουμε την μέθοδο διαμοιρασμού με βάση τη χωρική πληροφορία (spatial-first partitioning), κατά την όποια χωρίζουμε το σύνολο των χωροκειμενικών μας ερωτημάτων σύμφωνα με την γεωγραφική τους πληροφορία σε κελιά (Grid Partition Cells) κάτι που οδηγεί στην παράλληλη διεξαγωγή και επεξεργασία των ερωτημάτων. Για τον διαχωρισμό και την κατανομή των δεδομένων, υλοποιήθηκε ένας μηχανισμός δημιουργίας ενός ομοιόμορφου πλέγματος (Uniform Grid), το οποίο θα χωρίζει τον γεωγραφικό χάρτη σε παραλληλόγραμμες περιοχές/κελιά (Cells). Κάθε κελί είναι μη επικαλυπτόμενο από τα υπόλοιπα και περιέχει queries, $q \in Q$, και objects , $t \in T$, που ανήκουν στη γεωγραφική περιοχή που περικλείει. Το μέγεθος και ο αριθμός των κελιών του Grid είναι πλήρως παραμετροποιήσιμα μπορούν να καθοριστούν από τον χρήστη.



Εικόνα 11 *- Uniform Grid 4x4*

Στο παραπάνω γράφημα (βλ. Εικόνα 4.3) βλέπουμε ένα ομοιόμορφο πλέγμα (Uniform Grid) 16 κελιών. Τα αντικείμενα (Tweet objects) τα οποία καταφθάνουν συνεχώς απεικονίζονται με μπλε χρώμα ενώ τα στατικά ερωτήματα (Static Queries) με τα οποία θα πραγματοποιηθεί η ενδεχόμενη ζεύξη αναπαρίστανται με πράσινο χρώμα. Όταν θέλουμε να εξετάσουμε μια πιθανή ζεύξη μεταξύ κάποιου ερωτήματος (query) και κάποιου αντικειμένου (object), από το τελευταίο πακέτο πληροφορίας (mini-batch) στοχεύουμε στην εκτέλεση αυτών των εργασιών (Spark Jobs) εντός του κελιού και, κατ' επέκταση, εντός του ίδιου κόμβου (node) στο σύστημα (cluster). Έτσι κάνοντας χρήση της παράλληλης εκτέλεσης εργασιών επιτυγχάνουμε κατανομή του φόρτου εργασίας και αποφεύγουμε οποιαδήποτε καθυστέρηση που θα μπορούσε να υπάρξει λόγω της επικοινωνίας και της μεταφοράς δεδομένων μεταξύ των κόμβων του δικτύου.

```
// Input: num_of_cells (Number of Cells)
cells=[]
max_ver=90
min_ver=90-ver_len
min_hor=-180-hor_len
max_hor=-180
for c in range(1,num_of_cells+1):
    min_hor+=hor_len
    max_hor+=hor_len
    temp_cell = {"cell":c,
                 "max_lat": max_ver,
                 "min_lat": min_ver,
                 "min_lon": min_hor,
                 "max_lon": max_hor
                }
    cells.append(temp_cell)
    if max_hor==180:
        min_hor= -180-hor_len
        max_hor= -180
        max_ver-=ver_len
        min_ver-=ver_len
// Result: cells (The final cells of Uniform Grid)
```

Algorithm 1: Python Script for Creation of Uniform Grid

Ο αλγόριθμος, Algorithm 1, που παρουσιάζεται για την δημιουργία του ομοιόμορφου πλέγματος (Uniform Grid) που θα χρησιμοποιηθεί σαν μηχανισμός ευρετηρίασης (indexing) των χωροκειμενικών αντικειμένων έχει πολυπλοκότητα $O(N)$, όπου N το πλήθος των cells που ορίζονται ως είσοδο (initial parameters) του αλγορίθμου. Κάθε παραγόμενο cell, $c \in C$ που περιέχεται στη λίστα *cells*, του συνόλου C, έχει τη μορφή:

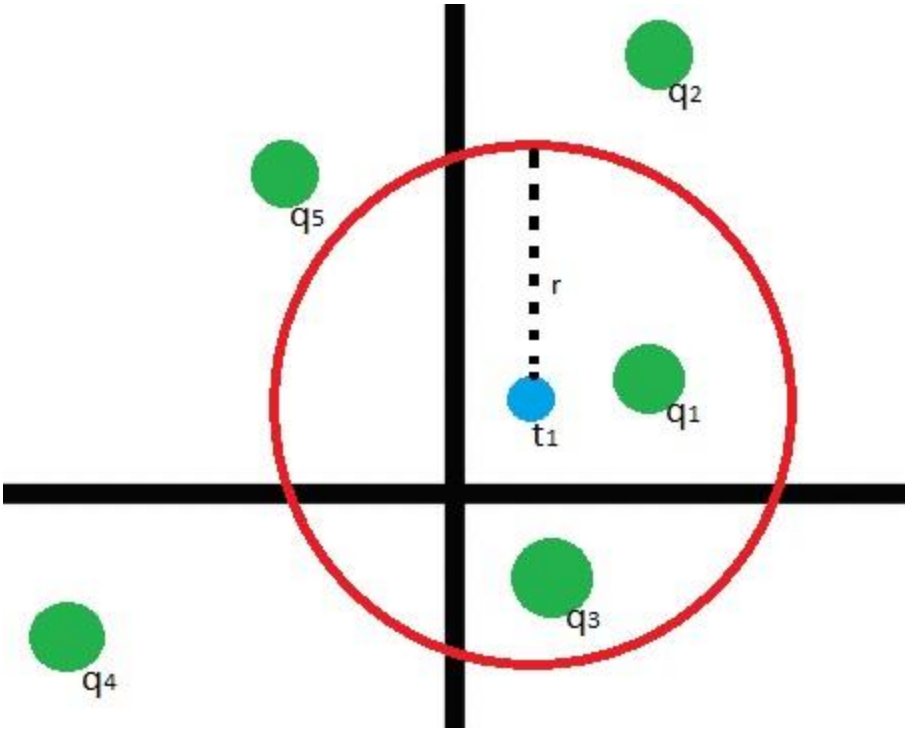
```
{
  'id': cell_id,
  'max_lat': NumValue,
  'max_lon': NumValue,
  'min_lat': NumValue,
  'min_lon': NumValue
}
```

Single Cell Schema

4.3.2 Διεξαγωγή ερωτημάτων και Duplication

Όπως αναφέρθηκε, θεωρούμε ένα σύνολο από χωροκειμενικά ερωτήματα (Query objects) Q, ως ένα στασικό σύνολο ερωτημάτων (static batch data) τα οποία θα κατανεμηθούν από τον μηχανισμό διαμοιρασμού (Grid Partitioning) και θα οδηγήσουν στις πιθανές ζεύξεις με τα tweet αντικείμενα της συνεχόμενης ροής. Στοχεύοντας στην ορθή αξιολόγηση των αποτελεσμάτων και αποδοτικότητας της εφαρμογής μας, χρησιμοποιούμε διαφορετικά σύνολα δεδομένων (Datasets) για τη δημιουργία του συνόλου Q ερωτημάτων. Έτσι, στα πλαίσια της πειραματικής μελέτης υλοποιήθηκε ένα script παραγωγής ψευδοτυχαίων (auto generated) $q \in Q$ ώστε να ακολουθούν διαφορετικές κατανομές στο χάρτη. Επιτρέποντας έτσι την αξιολόγηση του συστήματος τόσο σε περιπτώσεις όπου το σύνολο Q ερωτημάτων εκτείνεται σε όλο το εύρος του γεωγραφικού χάρτη και κατανέμεται ισόποσα στα ορισμένα κελιά αλλά και την αξιολογηση του συστήματος σε λιγότερο ισομοιρασμένες κατανομές των δεδομένων. Επίσης, αναπτύχθηκε η λειτουργία ενσωμάτωσης αυτών σε ένα αρχείο ή ακόμα και η σύνθεση αυτών από τον χρήστη, ώστε να συνάδει με κάθε περίπτωση που θέλουμε να εκτελέσουμε.

Τα ερωτήματα αυτά κατανέμονται κατά την διεργασία του Reduce στα worker nodes του cluster μας, ώστε να επιτρέπουν την κατανομή εργασιών (load balancing). Όπως αναφέρθηκε και προηγουμένως, η κατανομή αυτή γίνεται σύμφωνα με τη γεωγραφική πληροφορία και συγκεκριμένα, ανάλογα με τα κελιά (cells) στα οποία ανήκει κάθε αντικείμενο. Όταν ένα καινούργιο tweet καταφθάνει σε έναν κόμβο(node) σύμφωνα με το Cell identifier το οποίο και φέρει, θα πραγματοποιήσει ελέγχους ζεύξης με τα queries τα οποία βρίσκονται στο ίδιο worker node και φέρουν τον ίδιο cell identifier. Ένας cell identifier δίνεται σε κάποιο αντικείμενο (object) στην περίπτωση που η γεωγραφική πληροφορία ταυτίζεται με το γεωγραφικό πλαίσιο στο οποίο οριοθετείται το συγκεκριμένο κελί (Cell), $c \in C$. Η παραπάνω διαδικασία εξασφαλίζει πως κοντινά tweet objects και queries θα ελεγχθούν στο ίδιο worker node επιτυγχάνοντας την ελάχιστη καθυστέρηση λόγω της μεταφοράς δεδομένων και objects μεταξύ των κόμβων του συστήματος (cluster nodes). Ο χωρικός έλεγχος (spatial range) γίνεται συμφωνα με το range r, που έχει οριστεί ως είσοδος του συστήματος, πράγμα το οποίο μπορεί να οδηγήσει tweet objects τα οποία τοποθετούνται “κοντά” στα όρια ενός κελιού (Cell) να πρέπει να συζευχθούν με queries τα οποία βρίσκονται σε διαφορετικό κελί.



Εικόνα 12

Στο παράδειγμα που απεικονίζεται στην Εικόνα 12, το tweet object t_1ακολουθώντας την συνάρτηση $HavDist([q_{lon}, t_{lon}], [q_{lat}, t_{lat}]) \leq r$ πρέπει να πραγματοποιήσει Spatial Join με τα q_1, q_3καθώς είναι τα μόνα που βρίσκονται σε απόσταση μικρότερη του ορισμένου r. Μια παρατήρηση που αξίζει να σημειωθεί ειναι πως το q_2ερώτημα αν και βρίσκεται στο ίδιο cell με το t_1δεν αποτελεί αποτέλεσμα ζεύξης. Όσον αφορά στα δύο q με τα οποία δύναται να έρθει σε ζεύξη το αντικείμενο t_1 -αν φυσικά καλύπτονται και τα κριτήρια κειμενικής ομοιότητας- παρατηρείται ότι το q_1 ανήκει στο ίδιο cell με το t_1, ενώ το q_3ανήκει σε διαφορετικό γειτονικό cell.

Είναι πολύ σημαντικό, όπως έχει ήδη ειπωθεί, να ελαχιστοποιηθεί η μεταφορά δεδομένων μεταξύ των worker nodes. Μία προσέγγιση κατά την οποία κάθε $q \in Q$ θα περιέχεται μόνο στο cell, $c \in C$, το οποίο περικλείει τη γεωγραφική πληροφορία του q ($q.lon, qlat$) θα σηματοδοτούσε την ανάγκη για συνεχή μεταφορά της πληροφορίας των q των γειτονικών κελιών στα οποία διεξάγεται κάποιο Spatial Join ερώτημα με ένα tweet object. Κάτι τέτοιο θα οδηγούσε σε μια δραματική καθυστέρηση της ολοκλήρωσης των εργασιών και της γενικότερης αποδοτικότητας του συστήματος. Για να αποφευχθεί κάτι τέτοιο και για να διασφαλιστεί η απόδοση της διεξαγωγής των ερωτημάτων χωρικών και κειμενικών ζεύξεων, υλοποιήθηκε ένας μηχανισμός αντιγραφής ερωτημάτων (Query

Objects Duplication). Έτσι, σύμφωνα με το πλήθος και το μέγεθος των κελιών (Cells) και συνυπολογίζοντας τη δοθείσα απόσταση αναζήτησης (Searching Range), ένα Query, $q \in Q$, μπορεί να υπάρχει σε περισσότερα από ένα Cells διατηρώντας την χωροκειμενική του πληροφορία, ενώ ταυτόχρονα φέρει διαφορετικό Cell identifier. Αυτό μας δίνει τη δυνατότητα να αποθηκεύουμε κατάλληλα τα queries μας στους Workers του συστήματος αποτρέποντας κάθε αχρείαστη μεταφορά query ή tweet αντικειμένων μεταξύ των κόμβων του συστήματος. Το κριτήριο κατά το οποίο πραγματοποιείται αυτό το αντιγραφή (Duplication) της πληροφορίας βασίζεται στο r το οποίο έχει οριστεί και η αντιγραφή των $q_n \in Q$ μπορεί να γίνει σε κάθε γειτονικό cell, $c_1, c_2, \ldots, c_n \in C$, του εξεταζόμενου κελιού (original cell), c_n. Κάθε query μπορεί να αντιγραφεί έως και σε οκτώ κελιά, που είναι και ο μέγιστος αριθμός γειτονικών κελιών που μπορεί να υπάρξει. .

Για την ανάθεση του κάθε q στο original cell στο οποίο ανήκει, αλλά και για την υλοποίηση του duplication αναπτύχθηκε ο παρακάτω αλγόριθμος (βλ. Algorithm 2).

```
// Input: queries: list<JSON>, cells: list<JSON>, r: NumValue

final_queries = []
for q in queries:
    # Early termination flag
    flag = false
    while ( c in cells ) and ( flag==false ) :
        if ( q_belongs_to_c ):
            #  Assign c_id to curtain q
            q = {  'id': q.id,
                        'cell_id': c.id
                        'latitude': q.latitude,
                        'longitude': q.longitude,
                        'text': q.text
            }
            final_queries.append(q)
            flag = true

            #  Trigger Duplication mechanism for c
            duplicate_to_neighbor_cells_if_needed(q, r)
//  Result:  final_queries(The  final  queries  with  the  cell.id
identifier of the Uniform Grid and all the needed duplications)
```

Algorithm 2 - Duplicate Mechanism with Early Termination

Ο αλγόριθμος, Algorithm 2, δέχεται σαν ορίσματα την ακτίνα r, τη λίστα με τα queries, του συνόλου Q, καθώς και τη λίστα των cells, του συνόλου C, του Grid από τον Algorithm 1. Στη συνέχεια, για κάθε $q \in Q$ ελέγχουμε αν οι γεωγραφικές του συντεταγμένες είναι εντός του γεωγραφικού χώρου που καλύπτει ένα $c \in C$ ώστε να προστεθεί η cell id πληροφορία στο q και να δημιουργηθούν τα $q_{1,}\, q_{2,\dots,}\, q_n$ τα οποία είναι τα αντιγραμμένα ερωτήματα (duplicated queries) του q .

Ο Algorithm 2 ελέγχει αν σε ένα q έχει ήδη οριστεί ένα Cell Id, ενώ παράλληλα εμπεριέχει μία συνάρτηση πρόωρου τερματισμού (early termination) για τον υπολογισμό και την αντιγραφή του query στα γειτονικά κελιά ελαχιστοποιώντας του αναγκαίους ελέγχους.

Αποτέλεσμα είναι το σύνολο Q το οποίο περιέχει όλα τα q με το τελικό schema:

```
{
  'cell': cell_id,
  'id': query_id,
  'latitude': NumValue,
  'longitude': NumValue,
  'text': String
}
```

Single Query Object Final Schema

4.4 Υλοποίηση Συστήματος και Εκτέλεση

Όπως έχει ήδη αναφερθεί στα πλαίσια της εκτέλεσης του συστήματος και της διεξαγωγής των πειραματικών διαδικασιών, χρησιμοποιήθηκε μια Real-time ροή tweets τα οποία φέρουν χωροκειμενικό περιεχόμενο. Για την επίτευξη αυτού αναπτύξαμε ένα App κάνοντας χρήση της βιβλιοθήκης Tweepy και φιλτράροντας τα tweets τα οποία περιέχουν γεωγραφικές πληροφορίες. Στη συνέχεια, με χρήση ενός tcp socket τροφοδοτούμε τη ροή στην Apache Spark Streaming εφαρμογή μας.

4.4.1 Configuration Spark Streaming

Κατά την αρχικοποίηση (initialization) του Spark Streaming Context ορίζουμε τη δημιουργία micro-batches κάθε δέκα δευτερόλεπτα, ενώ παράλληλα ορίζουμε ένα κυλιόμενο παράθυρο (Sliding-window) της τάξεως των τριών λεπτών. Αυτό μας επιτρέπει να έχουμε μια εικόνα των αποτελεσμάτων και της κατανομής των αντικειμένων, καθώς και των συσχετίσεων μεταξύ τους για τα μικρό πακέτα πληροφορίας (micro-batches) που έλαβαν χώρα στο διάστημα αυτό.

4.4.2 Preprocess and Mapping

Για την ορθή υλοποίηση του μοντέλου MapReduce και της αποδοτικότερης κατανομής του φόρτου επεξεργασίας αποφασίστηκε να επεξεργαστούμε κάθε πακέτο (micro-batch) από tweets καθώς και τα ορισμένα ερωτήματα (Queries) ως Key/Value Pairs. Κατά τη διαδικασία του Mapping, στην περίπτωση των Queries, υλοποιούμε, τη μετατροπή αυτών σε Key/Value RDD έχοντας ως Key το Cell Identifier και ως Value τις τιμές, Query Id, Latitude, Longitude, Text, καθώς και ενός identifier που δηλώνει πως το συγκεκριμένο rdd είναι query object.

$$\forall\, q \in Q,\ q = (cell.id, (q.id,\ q.lat,\ q.lon,\ q.text,\ obj.type))$$

Κατά την έλευση κάθε πακέτου (micro-batch) tweet αντικειμένων, το Apache Spark Streaming Application δημιουργεί μια ακολουθία από Rdd objects ονόματι DStreams. Κρίθηκε απαραίτητη η μετατροπή τόσο των Queries όσο και των Tweets objects σε μία κοινή μορφή και schema ώστε να επιτραπεί το custom Reduce functionality το οποίο θα κάνει δυνατή την αποδοτικότερη κατανομή των δεδομένων μας.

Για κάθε tweet ορίζουμε ένα Key/Value Pair RDD, με αντίστοιχο schema, όπου $cell.id$ το κελί του Grid στο οποίο ανήκει, σύμφωνα με τη γεωγραφική του θέση την οποία και βλέπουμε στις τιμές Latitude και Longitude.

$$\forall\, t \in T,\ t = (cell.id, (t.author,\ t.lat,\ t.lon,\ t.text,\ obj.type))$$

Η τιμή $t.text$ που αποτελεί την κειμενική πληροφορία του tweet, το$obj.type$ το tweet object identifier, οι τιμές $t.lat$ και $t.lon$ τη γεωγραφική πληροφορία και η τιμή $t.author$ τον συγγραφέα του tweet.

4.4.3 Reduce and Distribution

Πριν τη διεξαγωγή του reduce, εκτελείται μία ένωση (union) των δύο όμοιων schemas Query και Tweet Key/Value Pair RDDs επιτρέποντας στον Custom Function Reducer να χρησιμοποιήσει το Key ως reduce αναφορά (reference) για την κατανομή στα partitions.

Κατά τη διεργασία του Reduce ο master orchestrator του συστήματος, κατανέμει σύμφωνα με το Cell Id τα objects στους workers όπου θα πραγματοποιηθούν και οι απαραίτητοι έλεγχοι για την ζεύξη και τη δημιουργία των τελικών αποτελεσμάτων όπως αυτά ζητήθηκαν. Αυτό επιτυγχάνεται με την υλοποίηση μιας custom reduceByKey συνάρτησης. Κατά την εκτέλεση της οποίας, τα Key/Value Pair Rdds που εμπεριέχονται στο ίδιο cell ελέγχονται τόσο σύμφωνα με το object identifier το οποίο φέρουν, όσο και με το αν επαληθεύουν την χωρική και κειμενική συνθήκη για την επίτευξη μιας ζεύξης. Παρακάτω παρατίθεται η ψευδογλώσσα των ελέγχων που χρησιμοποιήθηκαν για την παραγωγή ενός επιτυχούς join μεταξύ δύο objects που φέρουν τον ίδιο cell identifier, cell.id.

```
// Input: obj1: Rdd<Key,Value>, obj2: Rdd<Key,value>, r: NumValue, a:
NumValue

if (obj1.type != obj2.type):
    if haversine((obj1.lon,obj2.lon), (obj1.lat, obj2.lat)) <= r :
        if jaccard_rating(obj1_text, obj2.text) >= a:
            return (obj1.id, obj2.id)

// Return: SpatioTextual Similarity Join of a tweet object and a
query. (Tweet Author, query_id)
```

Spatiotextual Similarity Join

Κατά την ζεύξη κάθε tweet με ένα Query το οποίο περιέχει με τη σειρά του χωρική και κειμενική πληροφορία, χρησιμοποιείται η συνάρτηση haversine για το υπολογισμό της απόστασης. Παράλληλα, η κειμενική πληροφορία του tweet αντικειμένου και του Query επέρχεται επεξεργασίας κερματοποίησης (Tokenization) και την αφαίρεσης κοινών λέξεων (stopwords) και ειδικών χαρακτήρων. Έτσι αποδεικνύεται αποδοτικότερη η εφαρμογή της Jaccard similarity για την εξαγωγή του βαθμού ομοιότητας (similarity score) μέσω της οποίας θα επιτευχθεί και η τελική ζεύξη.

5. Πειραματική διαδικασία και Αποτελέσματα

Στο κεφάλαιο αυτό παρουσιάζεται η πειραματική μελέτη που ακολουθήθηκε, καθώς και μια γραφική απεικόνιση των αποτελεσμάτων που προέκυψαν από αυτή. Τα πειράματα και η εκτέλεση των υλοποιήσεων έγιναν σε ένα Standalone Cluster 5 nodes (ενός master και τεσσάρων workers) κατανέμοντας τη CPU (Intel i7) των 6 πυρήνων και 12 threads, καθώς και 16gb RAM. Παρακάτω περιγράφονται τα Datasets που χρησιμοποιήθηκαν, οι μετρικές αποτελεσματικότητας και απόδοσης καθώς και τα πειράματα που διεξήχθησαν.

5.1 Μετρικές αξιολόγησης συστήματος

- ### Αντικείμενα ανά Κελί (Objects per Cell)

  Βασική μετρική αξιολόγησης της αποτελεσματικότητας του partitioning μηχανισμού και του καταμερισμού εργασιών στα cluster nodes.

- ### Πλήθος Αντιγράφων

  Στόχος του συστήματός μας είναι η επίτευξη των σωστών αποτελεσμάτων χρησιμοποιώντας το μικρότερο δυνατό πλήθος αντιγράφων αντικειμένων μεταξύ των καθορισμένων κελιών του Uniform Grid.

- ### Χρόνος εκτέλεσης

  Χρησιμοποιούμε τον χρόνο εκτέλεσης όλου του pipeline του συστήματος για να αξιολογήσουμε την αποδοτικότητα εκτέλεσης και εφαρμογής του συνόλου των components και των μηχανισμών.

- ### Πλήθος συγκρίσεων

  Αξιολογούμε το σύστημα βάσει του αριθμού των απαιτούμενων συγκρίσεων που πρέπει να εφαρμοστούν ώστε να εξάγουμε τα επιθυμητά αποτελέσματα του Similarity Spatio Textual Join.

5.2 Datasets

Για την εκτέλεση της πειραματικής μελέτης και αξιολόγησης του συστήματος αναπτύξαμε ένα module το οποίο, όπως περιγράφηκε στο προηγούμενο κεφάλαιο, κάνει χρήση της βιβλιοθήκης Tweepy και του Twitter Stream API για τη συνεχόμενη συλλογή και διοχέτευση σε πραγματικό χρόνο tweet αντικειμένων στη Apache Spark Streaming εφαρμογή. Αυτή αποτελεί τη συνεχόμενη ροή χωροκειμενικών δεδομένων του

συστήματος και επιλέχθηκε με στόχο την αντικειμενικότητα που προσφέρει το real-time tweet streaming. Στη συνέχεια, για την ανάπτυξη της συλλογής των χωροκειμενικών ερωτημάτων επιλέχθηκε η ανάπτυξη τόσο ενός dataset, μεγέθους 1.583 γραμμών, αποτελούμενο από παρελθοντικά tweet αντικείμενα (Retrospective Tweet objects) τα οποία περιέχουν χωρική και κειμενική πληροφορία, αλλά και συνθετικών dataset με ελεγχόμενη γεωγραφική κατανομή των $q \in Q$ ερωτημάτων. Κατά την εκτέλεση των πειραμάτων αυξομειώσαμε τη γεωγραφική διασπορά των συνθετικών Datasets για την εξαγωγή πιο ολοκληρωμένων συμπερασμάτων ως προς τη συμπεριφορά του συστήματος.

5.2.1 Synthetic Datasets

Παρακάτω περιγράφονται τα βασικά χαρακτηριστικά των συνθετικών Datasets τα οποία περιέχουν τα Queries χωροκειμενικά ερωτήματα που χρησιμοποιούνται για τη ζεύξη με τη συνεχόμενη ροή tweets.

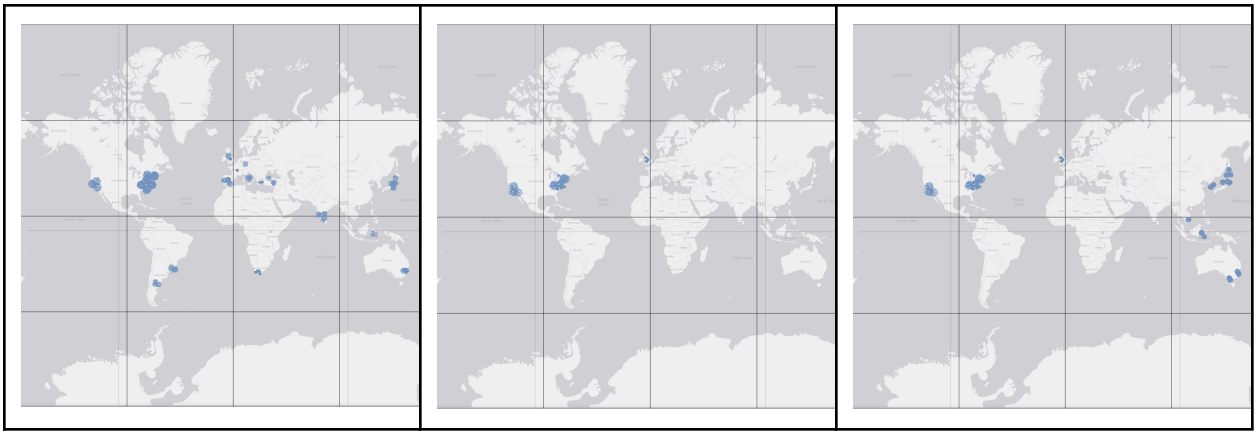
Schema

ID	Longitude	Latitude	Text
Integer	Double	Double	String

Πίνακας 2 *- Synthetic Datasets Schema*

Το schema που περιγράφεται στον παραπάνω πίνακα είναι το αρχικό της κάθε εγγραφής στο dataset των ερωτημάτων (Query Objects) πριν τον ανασχηματισμό απο τη διαδικασία διαμοιρασμού (Partitioning) και τον μετασχηματισμό κατά την εκτέλεση της εφαρμογής.

Distribution



Εικόνα 13 *- Datasets a, b , c*

Στην Εικόνα 13 βλέπουμε τις διαφορετικές γεωγραφικές κατανομές που χρησιμοποιούνται στα συνθετικά Datasets των Query αντικειμένων. Όπως παρατηρούμε το Dataset a είναι το πιο χωρικά ομοιόμορφο κατανεμημένο dataset. Όπως φαίνεται και στην Εικόνα 13 με χρήση 4x4 Uniform Grid κατανέμει τα 1.583 χωροκειμενικά αντικείμενά μοιρασμένα 7 κελιά. Λιγότερα ισομοιρασμένο είναι το Dataset c το οποίο με χρήση 4x4 Uniform Grid κατανέμει τα αντικείμενα του σε 4 κελιά. Τέλος χρησιμοποιήθηκε το Dataset b το οποίο μοιράζει τα αντικείμενα του σε μόλις 2 γειτονικά κελιά, με χρήση 4x4 Uniform Grid, ως παράδειγμα ανομοιόμορφης κατανομής.

Γίνεται εύκολα κατανοητό ότι μια ομοιόμορφη διασπορά των σημείων στο χάρτη είναι προτιμότερη από μια ανομοιόμορφη, καθώς με αυτόν τον τρόπο η κατανομή των αντικειμένων γίνεται ομοιόμορφα κατά το Spatial First partitioning στα κελιά του ομοιόμορφου πλέγματος. Αποφεύγονται κενά cells και επιτυγχάνεται η ορθότερη κατανομή εργασιών, συσχετίσεων, στους workers.

5.2.2 Real Time Stream

Παρακάτω περιγράφεται η πληροφορία που διοχετεύεται στη Spark Streaming εφαρμογή μας και χρησιμοποιείται ως χωροκειμενική συνεχόμενη ροή του συστήματος. Το Twitter Streaming API προσφέρει μια ροή από twitter json objects τα οποία περιέχουν πολύ μεγάλο πλήθος πληροφορίας. Χρησιμοποιούμε το component Data Handling και Parsing της ροής, κάνοντας το απαραίτητο filtering, aggregations, feature picking πριν διοχετεύσουμε την πληροφορία στην Spark Streaming εφαρμογή μας.

Schema

Tweet Author_Name	Longitude	Latitude	Text
String	Double	Double	String

Πίνακας 3 *- Streaming Data Schema*

Το τελικό feature schema πριν την εφαρμογή partitioning κατανομής και Spark Streaming εργασιών συσχέτισης είναι αυτο που περιγράφεται στον παραπάνω πίνακα.

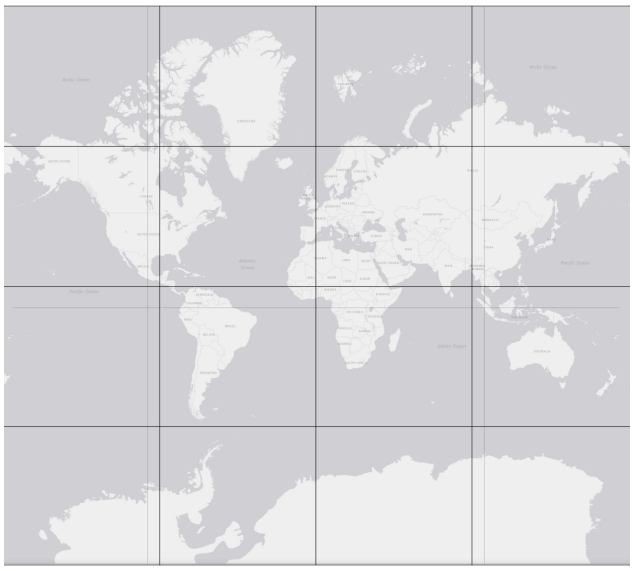
5.3 Πειραματική Μελέτη

Για την ορθή εξαγωγή συμπερασμάτων και αξιολόγηση της λύσης που παρουσιάζεται στην παρούσα έρευνα εκτελέσαμε ένα πλήθος πειραμάτων θέτοντας διαφορετικές τιμές εισόδου στις βασικές παραμέτρους του συστήματος. Οι βασικές παράμετροι εισόδου του συστήματος είναι το πλήθος των ορισμένων κελιών (Cells) του ομοιόμορφου πλέγματος (Uniform Grid) που υλοποιήθηκε για το spatial partitioning των δεδομένων, το μέγεθος της ακτίνας γεωγραφικής αναζήτησης r για την σύγκριση δύο σημείων με τη χρήση της Haversine function καθώς και το μέγεθος κειμενικής ομοιότητας που θα χρησιμοποιηθεί ως threshold α στην Jaccard Similarity. Παράλληλα, έγινε χρήση και παρουσίαση των αποτελεσμάτων για κάθε Query Dataset με σκοπό την καταγραφή της συμπεριφοράς του συστήματος σε διαφορετικές κατανομές δεδομένων.

Κατά την πειραματική διαδικασία χρησιμοποιήθηκαν τόσο πραγματικά δεδομένα (Retrospective Data) όσο και συνθετικά δεδομένα για την περιγραφή του συνόλου Q.

5.3.1 Πείραμα 1

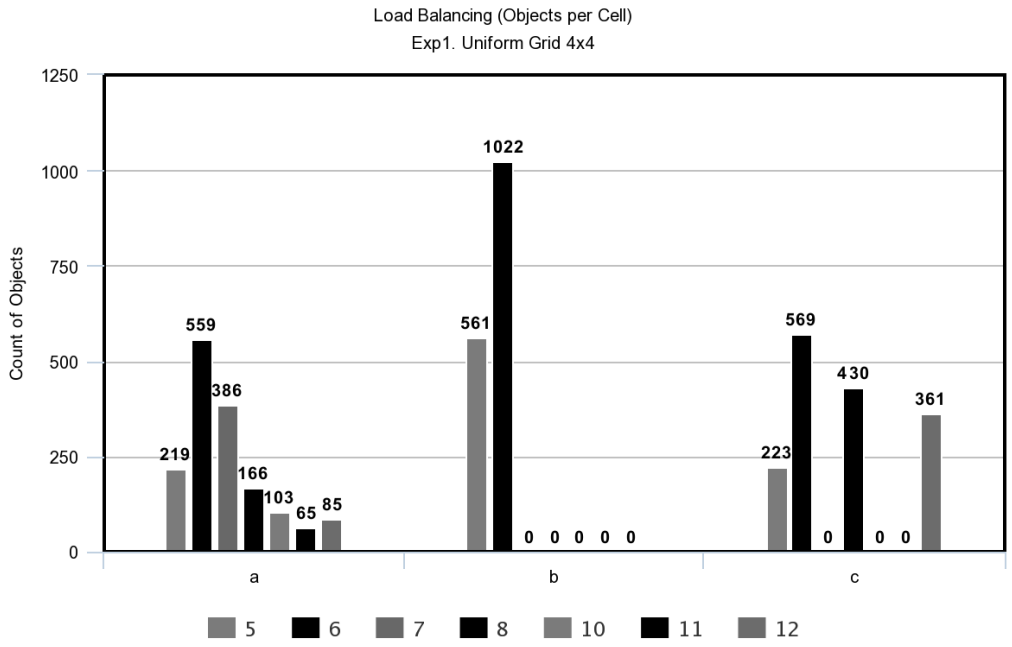
Στο Πείραμα 1 κάναμε χρήση ενός 4x4 Uniform Grid οπως φαίνεται στην Εικόνα 14 με μέγιστη ακτίνα αναζήτησης, r, τα 2501km. Γίνεται η χρήση των Query Datasets a, b, c για τις διαφορετικές παραμέτρους εισόδου Α, Β ,Γ.



Εικόνα 14 *- 4x4 Uniform Grid*

Αντικείμενα ανά Κελί

Στον άξονα x (x-axis) και στο υπόμνημα του Διαγράμματος (chart legend) 1 απεικονίζονται τα αποτελέσματα της εκτέλεσης των πειραμάτων μας για τα Query Datasets a, b, c (x-axis), οι κωδικοί των κελιών (Grid Cells IDs) και στον άξονα y το πλήθος των αντικειμένων (Count of Objects) που καταγράφεται σε αυτά ανά Dataset. Το διάγραμμα αυτό μας βοηθά να έχουμε μια εικόνα της κατανομής του φόρτου ανά κελί. Παρατηρούμε ότι το Dataset a με την ομοιόμορφη κατανομή παρουσιάζει καλύτερο διαμοιρασμό των αντικειμένων στο Uniform Grid σε αντίθεση με το Dataset b και c



Διάγραμμα 1 *- Load Balancing per Query Dataset (Objects per Cell)*

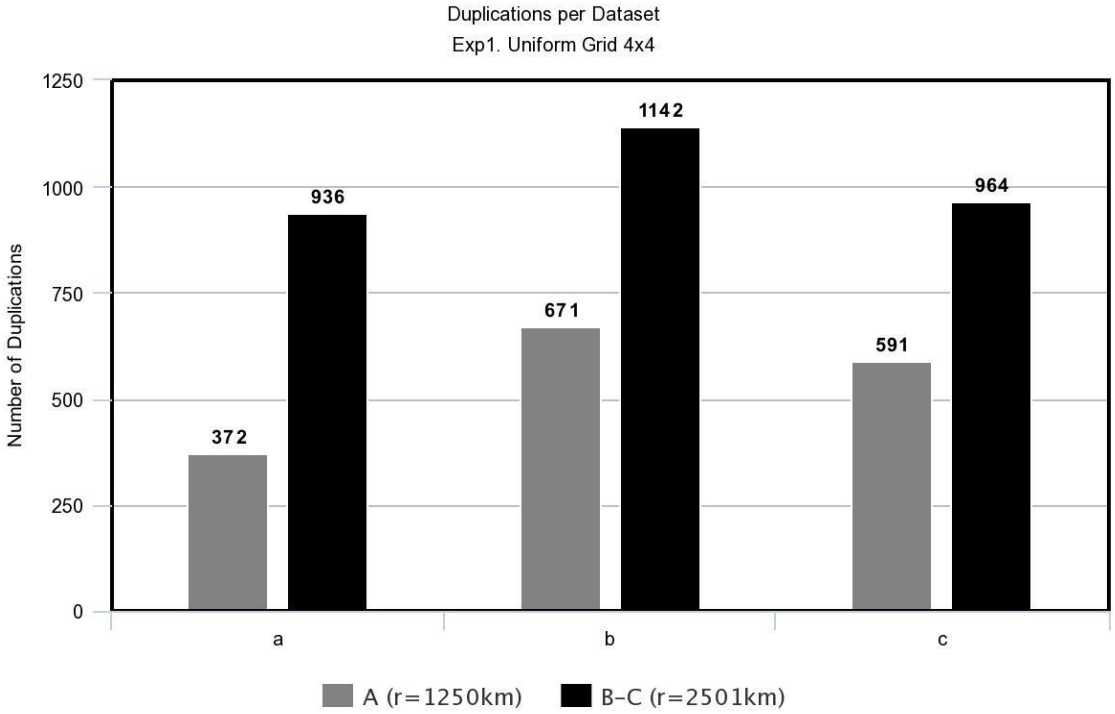
Πλήθος Αντιγραφών

Στη συνέχεια, διεξάγουμε το Πείραμα 1 χρησιμοποιώντας διαφορετικές τιμές εισόδου για τις παραμέτρους της ακτίνας αναζήτησης r και του threshold α της κειμενικής ομοιότητας.

Στον y-άξονα του Διαγράμματος 2 απεικονίζεται το πλήθος των αντιγραφών των αντικειμένων (Duplications) που πραγματοποιήθηκαν μεταξύ των κελιών ανά Query Dataset (x-άξονας), a, b ,c.

Οι τιμές εισόδου περιγράφονται στον παρακάτω πίνακα.

Execution	Number of Cells	Spatial Search Radius (r) (km)	Text Similarity Threshold (α)
A	16	1.250	0,2
B	16	2.501	0,8
Γ	16	2.501	0,2

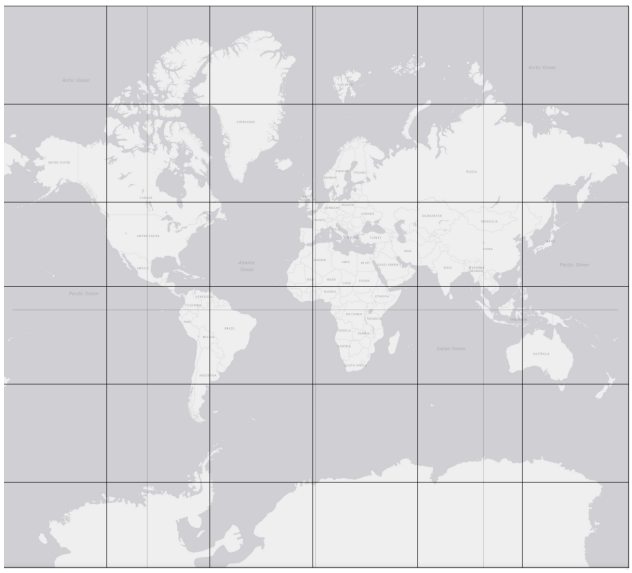


Διάγραμμα 2 *- Duplications per Execution and Dataset - Experiment 1*

Αυτό που παρατηρούμε μετά την πραγματοποίηση των εκτελέσεων Α, Β, Γ είναι μια αύξηση του πλήθους των αντιγράφων των αντικειμένων (Duplications) κατά την αύξηση της γεωγραφικής ακτίνας αναζήτησης r.

5.3.2 Πείραμα 2

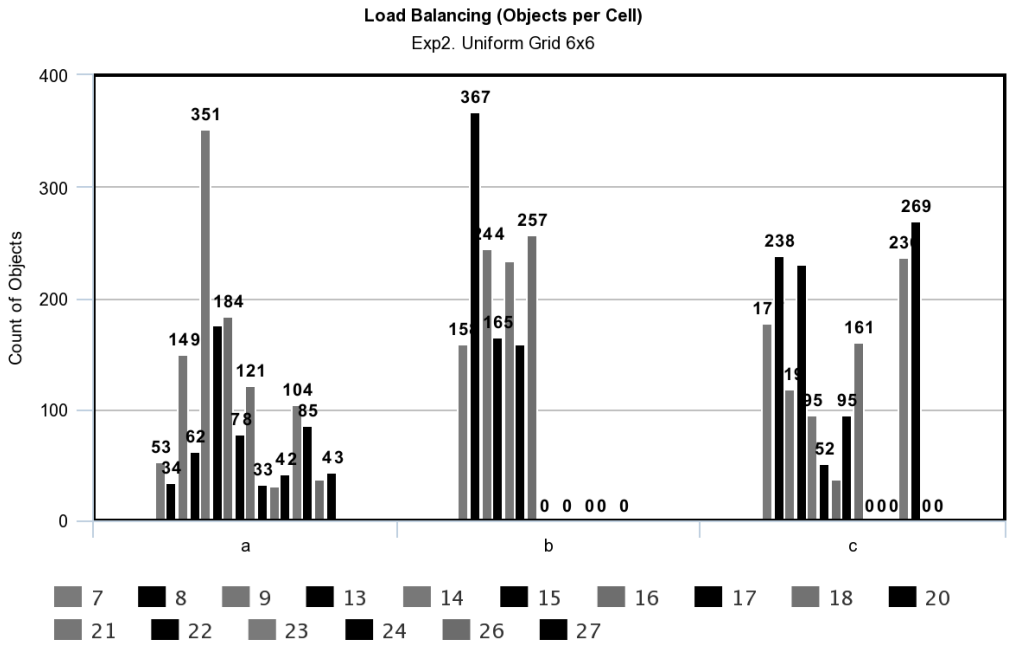
Στο Πείραμα 2 κάναμε χρήση ενός 6x6 Uniform Grid, οπως φαίνεται στην Εικόνα 15, με μέγιστη ακτίνα αναζήτησης, r, τα 1667km. Γίνεται χρήση των Query Datasets a, b, c για τις διαφορετικές παραμέτρους εισόδου A, B ,Γ.



Εικόνα 15 *- 6x6 Uniform Grid*

Αντικείμενα ανά Κελί

Στον άξονα x (x-axis) και στο υπόμνημα του Διαγράμματος (chart legend) 3 απεικονίζονται τα αποτελέσματα της εκτέλεσης των πειραμάτων μας για τα Query Datasets a, b, c (x-axis), οι κωδικοί των κελιών (Grid Cells IDs) και στον άξονα y το πλήθος των αντικειμένων (Count of Objects) που καταγράφεται σε αυτά ανά Dataset. Το διάγραμμα αυτό μας βοηθά να έχουμε μια εικόνα της κατανομής του φόρτου ανά κελί.



Διάγραμμα 3 *- Load Balancing per Query Dataset (Objects per Cell)*

Παρατηρούμε πως στο Διάγραμμα 3 σε σχέση με το αντίστοιχο διάγραμμα απεικόνισης κατανομής φόρτου του προηγούμενου πειράματος (βλ. Διάγραμμα 1), επήλθε μια εξισορρόπηση της κατανομής των αντικειμένων ανά κελί ανεξαρτήτως κατανομής και μεγέθους των Query Datasets εκτέλεσης a, b, c. Κάτι τέτοιο ήταν αναμενόμενο μετά την αύξηση του πλήθους των κελιών του Grid.

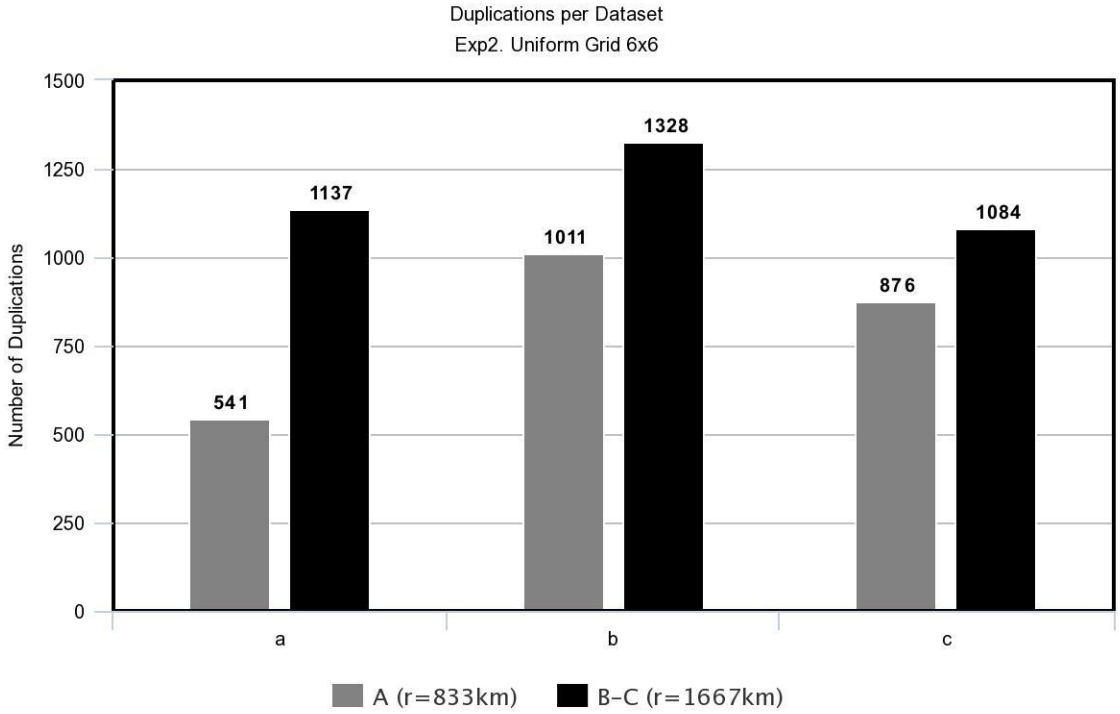
Πλήθος Αντιγραφών

Στη συνέχεια, διεξάγουμε το Πείραμα 2 χρησιμοποιώντας διαφορετικές τιμές εισόδου για τις παραμέτρους της ακτίνας αναζήτησης r και του threshold α της κειμενικής ομοιότητας.

Στον y-άξονα του Διαγράμματος 4 απεικονίζεται το πλήθος των αντιγραφών των αντικειμένων (Duplications) που πραγματοποιήθηκαν μεταξύ των κελιών ανά Query Dataset (x-άξονας), a, b ,c.

Οι τιμές εισόδου περιγράφονται στον παρακάτω πίνακα.

Execution	Number of Cells	Spatial Search Radius (r) (km)	Text Similarity Threshold (α)
A	36	833	0,2
B	36	1.667	0,2
Γ	36	1.667	0,8

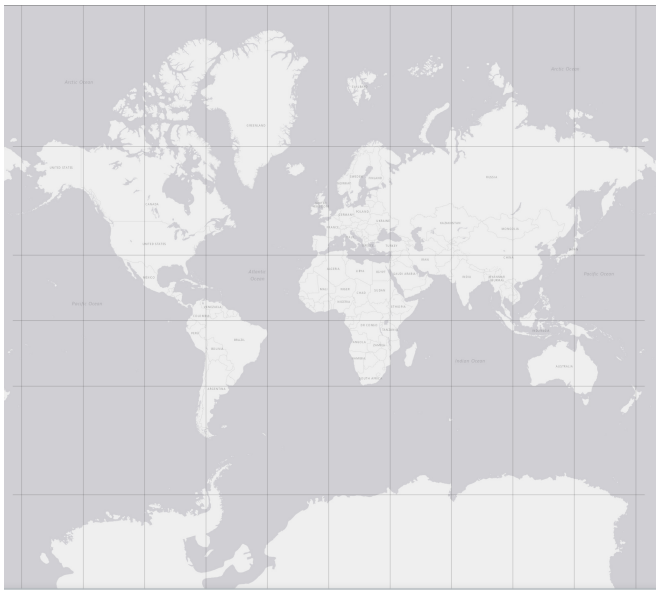


Διάγραμμα 4 *- Duplications per Execution and Dataset - Experiment 2*

Όπως ήταν αναμενόμενο βάσει των αποτελεσμάτων του προηγούμενου πειράματος, παρατηρούμε και εδώ αύξηση στο πλήθος των αντιγράφων των αντικειμένων (Duplications) κατά την αύξηση της γεωγραφικής ακτίνας αναζήτησης r.

5.3.3 Πείραμα 3

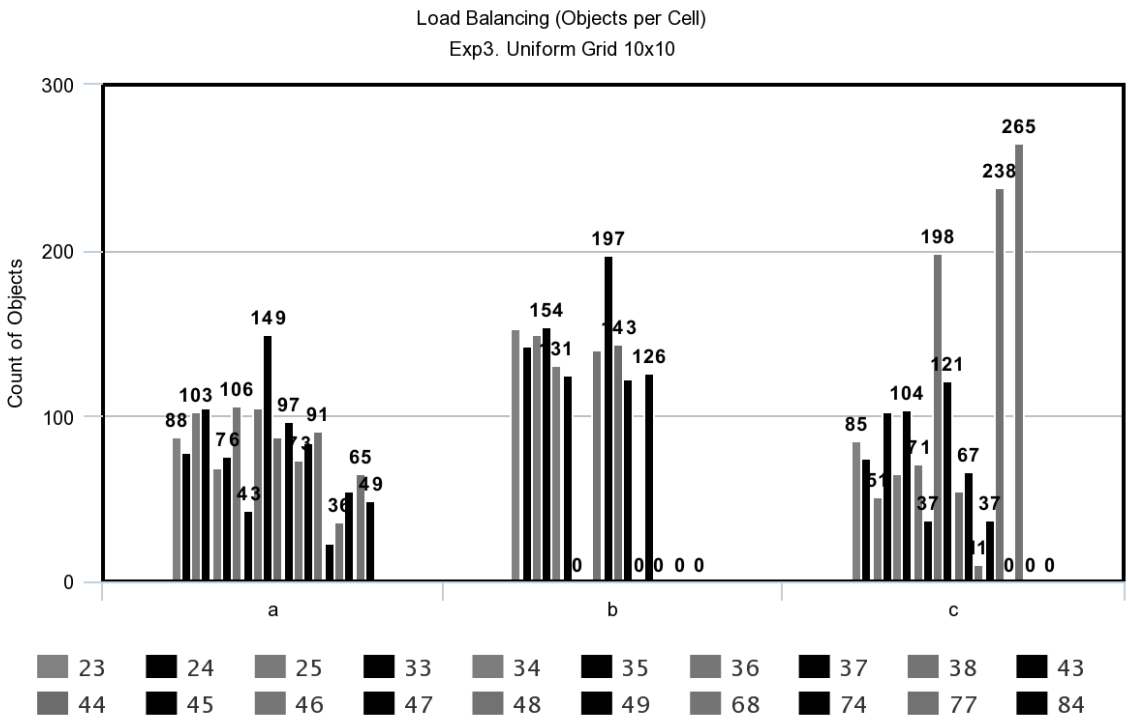
Στο Πείραμα 3 κάναμε χρήση ενός 10x10 Uniform Grid, οπως φαίνεται στην Εικόνα 16, με μέγιστη ακτίνα αναζήτησης, r, τα $1000km$. Γίνεται χρήση των Query Datasets a, b, c για τις διαφορετικές παραμέτρους εισόδου Α, Β ,Γ.



Εικόνα 16 - 10x10 Uniform Grid

Αντικείμενα ανά Κελί

Στον άξονα x (x-axis) και στο υπόμνημα του Διαγράμματος (chart legend) 5 απεικονίζονται τα αποτελέσματα της εκτέλεσης των πειραμάτων μας για τα Query Datasets a, b, c (x-axis), οι κωδικοί των κελιών (Grid Cells IDs) και στον άξονα y το πλήθος των αντικειμένων (Count of Objects) που καταγράφεται σε αυτά ανά Dataset. Το διάγραμμα αυτό μας βοηθά να έχουμε μια εικόνα της κατανομής του φόρτου ανά κελί.



Διάγραμμα 5 *- Load Balancing per Query Dataset (Objects per Cell)*

Παρατηρούμε πως στο Διάγραμμα 5 σε σχέση με τα αντίστοιχα διαγράμματα απεικόνισης κατανομής φόρτου των προηγούμενου πειραμάτων , Διαγράμματα 1 και 2, επήλθε μια εξισορρόπηση της κατανομής των αντικειμένων ανά κελί ανεξαρτήτως κατανομής και μεγέθους των Query Datasets εκτέλεσης a, b, c. Κάτι τέτοιο ήταν αναμενόμενο μετά την αύξηση του πλήθους των κελιών του Grid.

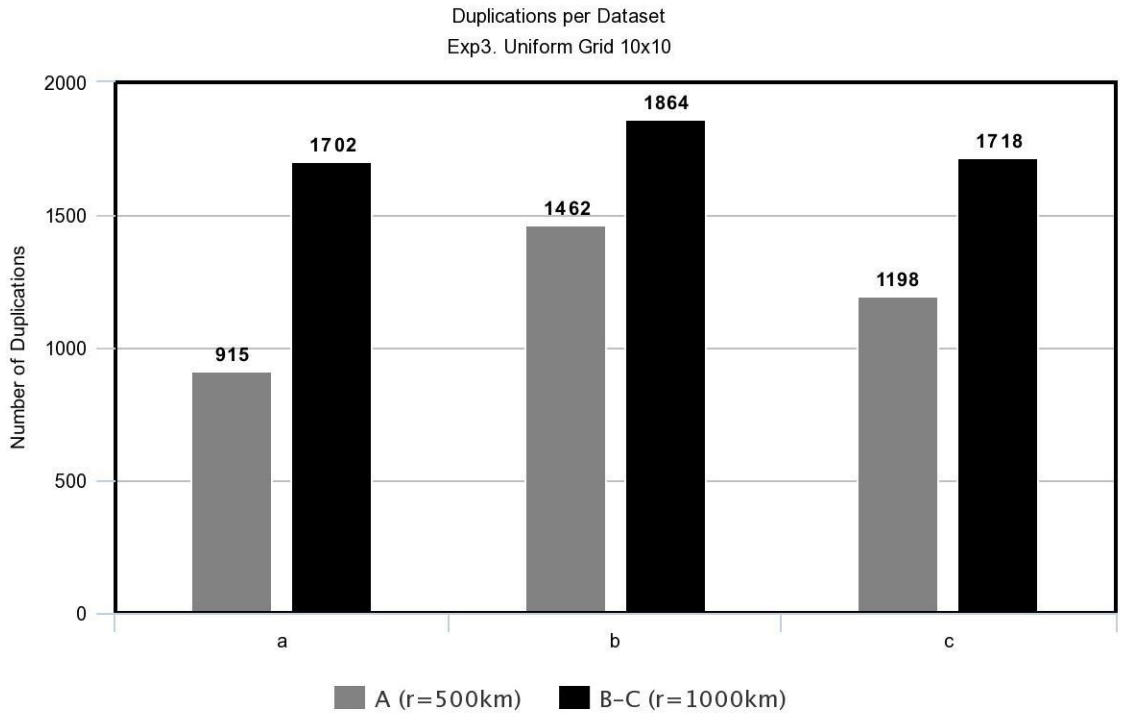
Πλήθος Αντιγραφών

Στη συνέχεια, διεξάγουμε το Πείραμα 3 χρησιμοποιώντας διαφορετικές τιμές εισόδου για τις παραμέτρους της ακτίνας αναζήτησης r και του threshold α της κειμενικής ομοιότητας.

Στον y-άξονα του Διαγράμματος 6 απεικονίζεται το πλήθος των αντιγραφών των αντικειμένων (Duplications) που πραγματοποιήθηκαν μεταξύ των κελιών ανά Query Dataset (x-άξονας), a, b ,c.

Οι τιμές εισόδου περιγράφονται στον παρακάτω πίνακα.

Execution	Number of Cells	Spatial Search Radius (r) (km)	Text Similarity Threshold (α)
A	100	500	0,2
B	100	1000	0,2
Γ	100	1000	0,8

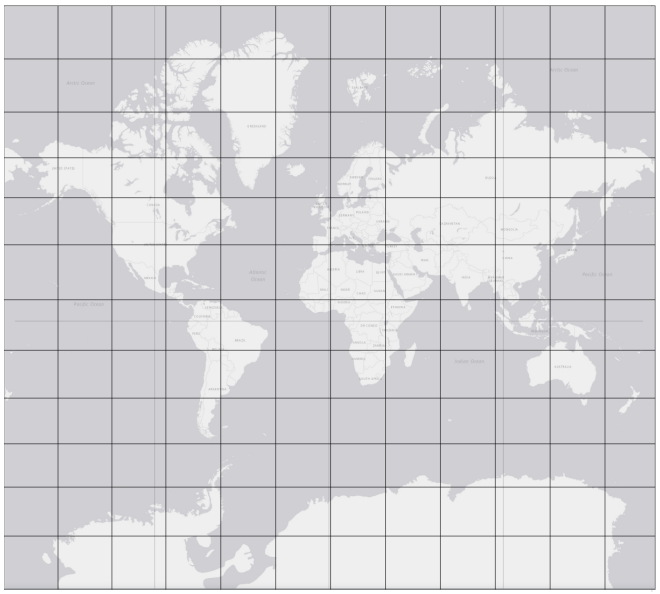


Διάγραμμα 6 *- Duplications per Execution and Dataset - Experiment 3*

Όπως και στις δύο προηγούμενες δοκιμές, έτσι και σε αυτήν, αυτό που παρουσιάζει αύξηση είναι το πλήθος των αντιγράφων αντικειμένων (Duplications) κατά την αύξηση της γεωγραφικής ακτίνας αναζήτησης r.

5.3.4 Πειραμα 4

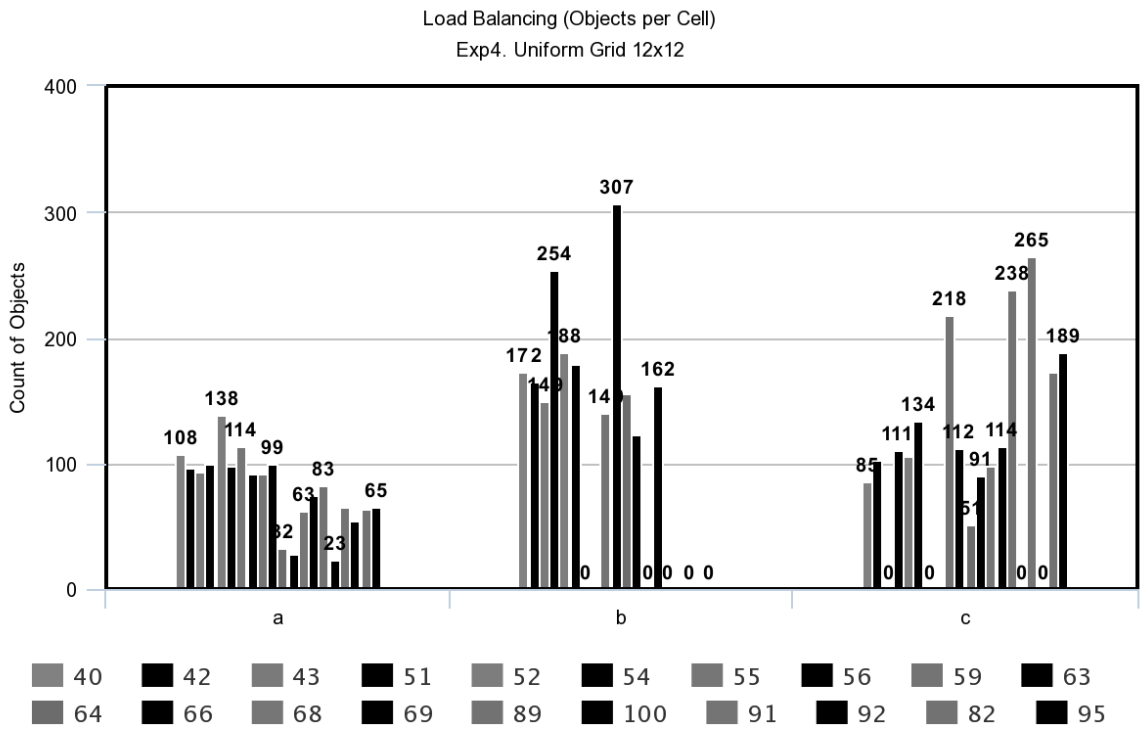
Στο Πειραμα 4 κάναμε χρήση ενός 12x12 Uniform Grid, οπως φαίνεται στην Εικόνα 17, με μέγιστη ακτίνα αναζήτησης, r, τα 833km.



Εικόνα 17 *- 12x12 Uniform Grid*

Στον άξονα x (x-axis) και στο υπόμνημα του Διαγράμματος (chart legend) 7 απεικονίζονται τα αποτελέσματα της εκτέλεσης των πειραμάτων μας για τα Query Datasets a, b, c (x-axis), οι κωδικοί των κελιών (Grid Cells IDs) και στον άξονα y το πλήθος των αντικειμένων (Count of Objects) που καταγράφεται σε αυτά ανά Dataset. Το διάγραμμα αυτό μας βοηθά να έχουμε μια εικόνα της κατανομής του φόρτου ανά κελί.

Αντικείμενα ανά Κελί



Διάγραμμα 7 - Load Balancing 12x12

Παρατηρούμε πως το Διάγραμμα 7 σε σχέση με τα αντίστοιχα διαγράμματα απεικόνισης κατανομής φόρτου των προηγούμενου πειραμάτων , Διαγράμματα 1, 3 και 5, επήλθε μια εξισορρόπηση της κατανομής των αντικειμένων ανά κελί ανεξαρτήτως κατανομής και μεγέθους των Query Datasets εκτέλεσης a, b, c. Κάτι τέτοιο ήταν αναμενόμενο μετά την αύξηση του πλήθους των κελιών του Grid.

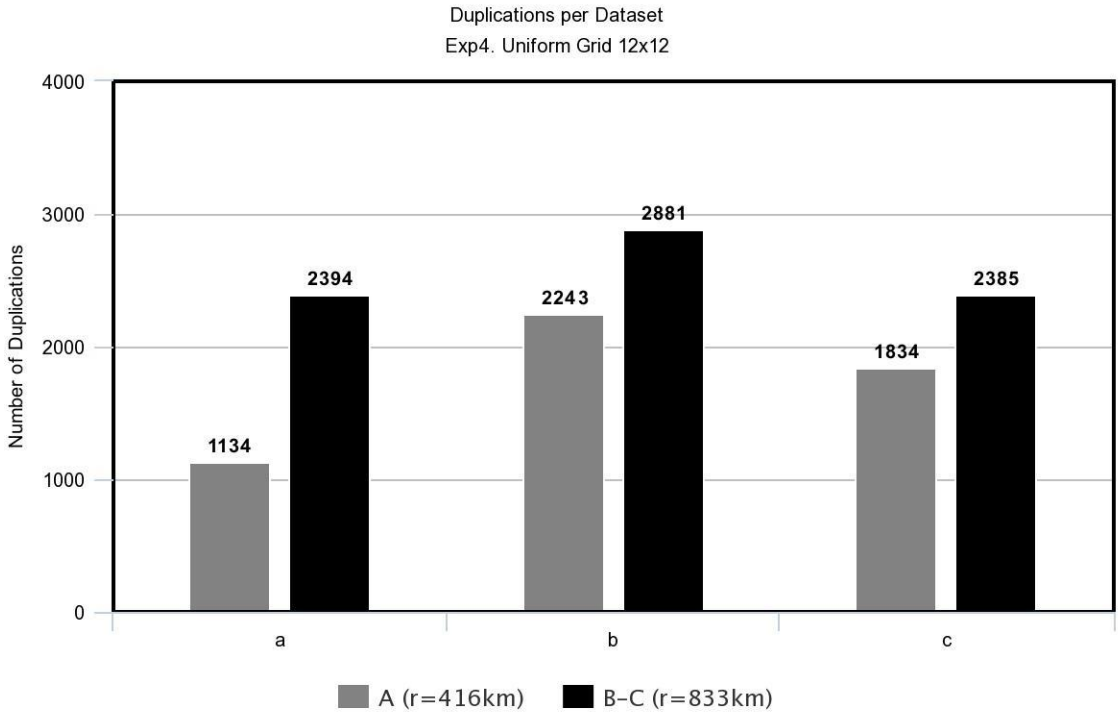
Πλήθος Αντιγραφών

Στη συνέχεια, διεξάγουμε το Πείραμα 4 χρησιμοποιώντας διαφορετικές τιμές εισόδου για τις παραμέτρους της ακτίνας αναζήτησης r και του threshold α της κειμενικής ομοιότητας.

Στον y-άξονα του Διαγράμματος 8 απεικονίζεται το πλήθος των αντιγραφών των αντικειμένων (Duplications) που πραγματοποιήθηκαν μεταξύ των κελιών ανά Query Dataset (x-άξονας), a, b ,c.

Οι τιμές εισόδου περιγράφονται στον παρακάτω πίνακα.

Execution	Number of Cells	Spatial Search Radius (r) (km)	Text Similarity Threshold (α)
A	144	416	0,2
B	144	833	0,2
Γ	144	833	0,8



Διάγραμμα 8 *- Duplications per Execution and Dataset - Experiment 4*

Τέλος, όπως ήταν αναμενόμενο, η αύξηση του πλήθους των αντιγράφων αντικειμένων (Duplications) που παρατηρήθηκε σε όλες τις προηγούμενες δοκιμές , κατά την αύξηση της γεωγραφικής ακτίνας αναζήτησης r, επαληθεύεται και στο 4ο πείραμα.

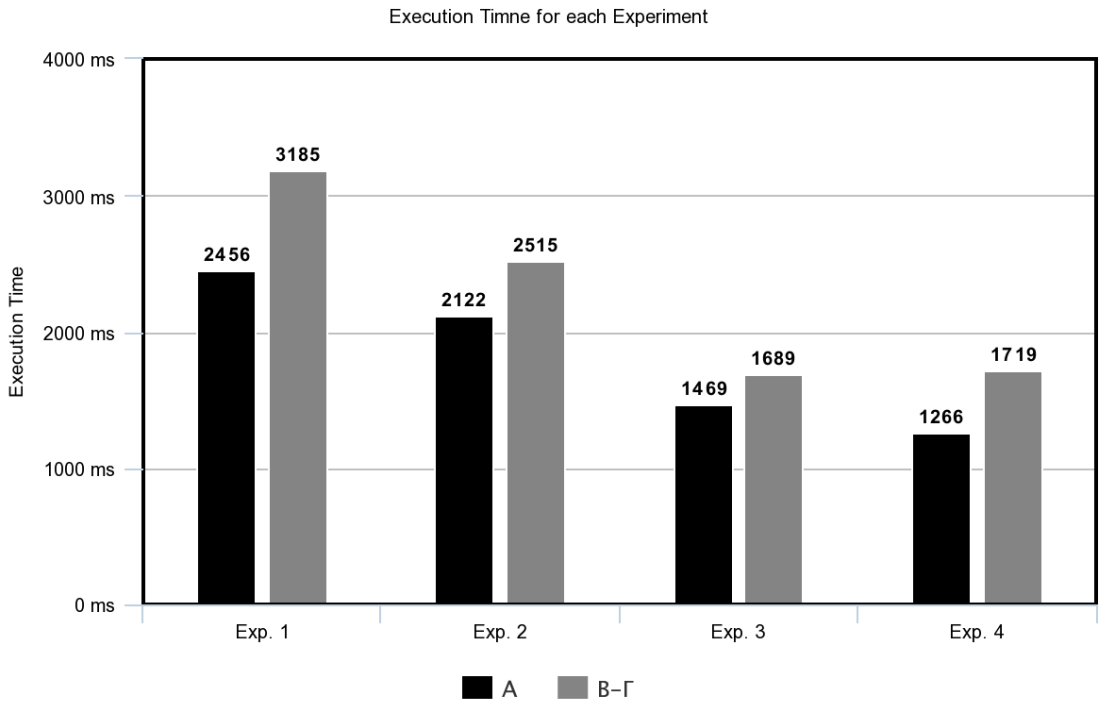
5.3.5 Αποτελέσματα

Μετά την εκτέλεση των πειραμάτων, εξήχθησαν και καταγράφηκαν κάποια συμπεράσματα σχετικά με τη συμπεριφορά της εφαρμογής ανάλογα με τις διαφορετικές τιμές εισόδου.

Όπως παρατηρείται και στα παραπάνω διαγράμματα των πειραματικών διαδικασιών, η αύξηση του πλήθους των κελιών του Uniform Grid επιφέρει καλύτερο διαμοιρασμό των δεδομένων ανεξαρτήτως πολυπλοκότητας και ομοιόμορφης ή μη κατανομής των Query objects.

Σημαντική κρίνεται η ραγδαία αύξηση του αριθμού των duplication των Query objects μεταξύ των κελιών κάτι το οποίο γίνεται εμφανέστερο για τις εκτελέσεις Β και Γ, όπου το search radius distance, $r\ distance$, φέρει τη μέγιστη τιμή για κάθε πείραμα.

Στο παρακάτω διάγραμμα περιγράφεται η μέση ταχύτητα εκτέλεσης, y-άξονας, των ζεύξεων κάθε πακέτο tweet αντικειμένων (micro-batch) σε διάστημα μιας ώρας για κάθε πείραμα, x-άξονας, με τη χρήση των διαφορετικών τιμών και παραμέτρων εισόδου όπως αυτά παρουσιάζονται στο υπόμνημα (legend).

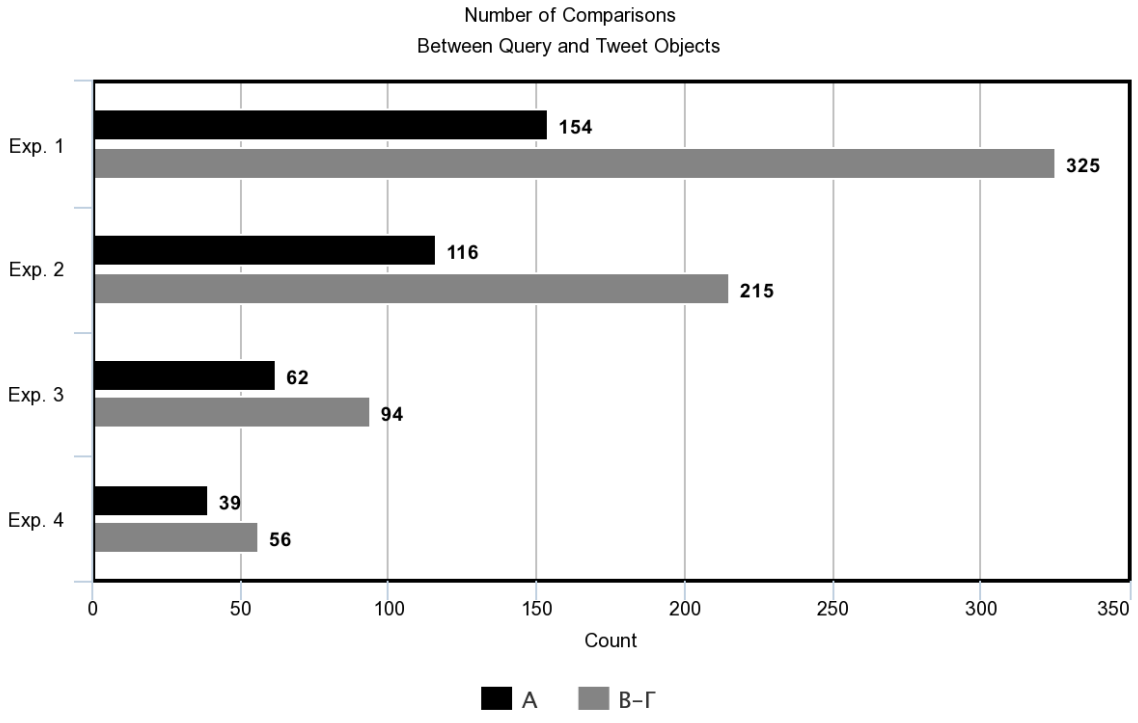


Διάγραμμα 9 *- Experiments Execution Time*

Παρατηρούμε πως ο χρόνος εκτέλεσης μειώνεται καθώς αυξάνεται το πλήθος των κελιών διαμοιρασμού. Πιο συγκεκριμένα, από τα διαγράμματα κατανομής φόρτου μπορούμε να συμπεράνουμε πως για τα Πειράματα 3 και 4 όπου ο αριθμός των Partition Cells ήταν 100 και 144 αντίστοιχα, ο διαμοιρασμός της πληροφορίας είναι πιο ομοιόμορφος κάτι που βοηθά στην αποδοτική εκτέλεση των διαδικασιών MapReduce και εν συνεχεία στην διεξαγωγή των τελικών ζεύξεων.

Αξίζει να σημειωθεί, ωστόσο, πως ενώ το Πείραμα 4 με τα 144 Partitions Cells μοιάζει να είναι το αποδοτικότερο σε όλες τις κατηγορίες Dataset, τιμές spatial search radius και text similarity thresholds, παρόλα αυτά, δεν φαίνεται να είναι απόλυτα σωστή αυτή η εκτίμηση. Ο χρόνος εκτέλεσης των ζεύξεων δείχνει να αυξάνεται σε σχέση με αυτόν του Πειράματος 3 για τις μέγιστες τιμές των τιμών αυτών, γεγονός το οποίο οφείλεται σε μεγάλο βαθμό στον αυξημένο αριθμό αντιγράφων αντικειμένων (Duplication Objects) (βλ. Διάγραμμα 9).

Ο x-άξονας του επόμενου διαγράμματος (βλ. Διάγραμμα 10) απεικονίζει το μέσο πλήθος συσχετίσεων και διεξαχθέντων ελέγχων μεταξύ των Query και Tweet Objects ανά κελί σε κάθε ένα από τα τέσσερα πειράματα, όπως βλέπουμε στον y-άξονα. Στο υπόμνημα του διαγράμματος αναγράφονται οι διαφορετικές τιμές εισόδου εκτέλεσης του εκάστοτε πειράματος.



Διάγραμμα 10 *- Number of Comparison for each Experiment*

Στο παραπάνω διάγραμμα παρατηρούμε την υπεροχή των πειραμάτων 3 και 4 και την εμφανή μείωση του αριθμού των συγκρίσεων μεταξύ των αντικειμένων καθώς αυξάνεται ο αριθμός των Partition Cells. Η παρατήρηση αυτή μεταφράζεται σε βελτίωση της κατανομής του φόρτου εργασίας (Load Balancing) και στην συνολικά αποδοτικότερη εκτέλεση των διεργασιών.

6. Συμπεράσματα και σκέψεις για μελλοντική μελέτη

Συνοψίζοντας, στα πλαίσια της παρούσας ερευνητικής εργασίας μελετήθηκε η βιβλιογραφία και συγκεντρώθηκαν οι ανάγκες, οι τεχνικές και τα εργαλεία που συνδέονται με το πεδίο της επεξεργασίας και της εκτέλεσης συσχετίσεων σε συνεχόμενες ροές χωροκειμενικών δεδομένων. Έγινε μελέτη των καλύτερων τακτικών και υπαρχουσών λύσεων με στόχο την υιοθέτηση αυτών.

Σχεδιάστηκε και αναπτύχθηκε ένα σύστημα επεξεργασίας και διεξαγωγής Spatio-Textual Similarity Joins χωροκειμενικών αντικειμένων από μία συνεχόμενη ροή πραγματικού χρόνου σε micro-batches κάνοντας χρήση των δυνατοτήτων του Spark Streaming και εφαρμόζοντας τεχνικές MapReduce και Spatial First Partitioning σε Uniform Grid. Υλοποιήθηκαν τεχνικές επεργασία και μετασχηματισμού των stream και static δεδομένων καθώς και η ανάπτυξη των απαραίτητων διεργασιών για την αποδοτικότερη κατανομή πληροφορίας και φόρτου εργασίας.

Για την πειραματική διαδικασία έγινε χρήση synthetic και retrospective datasets για την σύνθεση ερωτημάτων επαλήθευσης και του Tweet Live API για την εισαγωγή της συνεχόμενης ροής στην υλοποιηθείσα εφαρμογή και πραγματοποιήθηκαν πειράματα με διαφορετικές τιμές εισόδου.

Μέσω της κατανεμημένης εφαρμογής που αναπτύχθηκε, αξιοποιώντας τα οφέλη που προσφέρει το εργαλείο του Apache Spark Streaming, καταφέραμε να εξάγουμε πολύ θετικά αποτελέσματα. Παρατηρήσαμε πολύ καλή κατανομή του φόρτου εργασίας και μείωση των χρόνων εκτέλεσης καθώς αυξάνεται το πλήθος των κελιών του ομοιομορφου πλέγματος. Παράλληλα, η χρήση μηχανισμού αντιγραφής (Duplication) ελαχιστοποίησε την αχρείαστη μετακίνηση αντικειμένων μεταξύ των κελιών (Grid Cells), αποτρέποντας ταυτόχρονα το κώλυμα των εργασιών (Bottleneck). Ο μετασχηματισμός των δεδομένων και η εφαρμογή των τεχνικών MapReduce σε κάθε microbatch εισερχόμενων και στατικών αντικειμένων, συνέβαλαν σημαντικά στην επίτευξη των τελικών αποτελεσμάτων, όπως αυτά περιγράφονται εκτενώς στην Πειραματική Μελέτη του Κεφαλαίου 5.

Σαν μελλοντική πρόταση θα μπορούσε να αναπτυχθεί ένας Real-Time adjustable μηχανισμός partitioning που θα λειτουργεί σαν dynamic load-balancing των δεδομένων στο διάστημα που μεσολαβεί μεταξύ της έλευσης διαφορετικών micro-batches της ροής. Σαν δεύτερη πρόταση θα μπορούσε να πραγματοποιηθεί μελέτη για την ανάπτυξη ενός text-oriented ή spatial-text hybrid μηχανισμού partitioning και κατανομής της πληροφορίας.

Βιβλιογραφία

[1]Ahmed R. Mahmood, Ahmed M. Aly, Thamir Qadah, El Kindi Rezig, Anas Daghistani, Amgad Madkour, Ahmed S. Abdelhamid, Mohamed S. Hassan, Walid G. Aref, Saleh M. Basalamah:
Tornado: A Distributed Spatio-Textual Stream Processing System. Proc. VLDB Endow. 8(12): 2020-2023 (2015)

[2]Ahmed R. Mahmood, Anas Daghistani, Ahmed M. Aly, MingJie Tang, Saleh M. Basalamah, Sunil Prabhakar, Walid G. Aref:
Adaptive processing of spatial-keyword data over a distributed streaming cluster. SIGSPATIAL/GIS 2018: 219-228

[3]Xiang Wang, Ying Zhang, Wenjie Zhang, Xuemin Lin, Zengfeng Huang:
SKYPE: Top-k Spatial-keyword Publish/Subscribe Over Sliding Window. Proc. VLDB Endow. 9(7): 588-599 (2016)

[4]Zhida Chen, Gao Cong, Zhenjie Zhang, Tom Z. J. Fu, Lisi Chen:
Distributed Publish/Subscribe Query Processing on the Spatio-Textual Data Stream. ICDE 2017: 1095-1106

[5]Ahmed R. Mahmood, Walid G. Aref:
Scalable Processing of Spatial-Keyword Queries. Synthesis Lectures on Data Management, Morgan & Claypool Publishers 2019

[6]Anas Daghistani, Walid G. Aref, Arif Ghafoor, Ahmed R. Mahmood:
SWARM: Adaptive Load Balancing in Distributed Streaming Systems for Big Spatial Data. CoRR abs/2002.11862 (2020)

[7]Yue Chen, Zhida Chen, Gao Cong, Ahmed R. Mahmood, Walid G. Aref:
SSTD: A Distributed System on Streaming Spatio-Textual Data. Proc. VLDB Endow. 13(11): 2284-2296 (2020)

[8]Maryam Ghafouri, Xiang Wang, Long Yuan, Ying Zhang, Xuemin Lin:
Maintaining Boolean Top-K Spatial Temporal Results in Publish-Subscribe Systems. ADC 2018: 147-160

[9]Xiang Wang:
Efficient publish/subscribe processing over geo-textual stream. University of New South Wales, Sydney, Australia, 2017

[10]Ahmed R. Mahmood, Ahmed M. Aly, Walid G. Aref:
FAST: Frequency-Aware Indexing for Spatio-Textual Data Streams. ICDE 2018: 305-316

[11]Xin Jin, Zhengyi Yang, Xuemin Lin, Shiyu Yang, Lu Qin, You Peng:
FAST: FPGA-based Subgraph Matching on Massive Graphs. CoRR abs/2102.10768 (2021)

[12]Jianye Yang, Wenjie Zhang, Xiang Wang, Ying Zhang, Xuemin Lin:
Distributed Streaming Set Similarity Join. ICDE 2020: 565-576

[13]Jeffrey Dean, Sanjay Ghemawat:
MapReduce: Simplified Data Processing on Large Clusters. OSDI 2004: 137-150