ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
Μεταπτυχιακό Πρόγραμμα Σπουδών
"Πληροφοριακά Συστήματα & Υπηρεσίες"

UNIVERSITY OF PIRAEUS
DEPARTMENT OF DIGITAL SYSTEMS
Postgraduate Programme
"Information Systems & Services"

ΚΑΤΕΥΘΥΝΣΗ: Μεγάλα Δεδομένα και Αναλυτική

Επεξεργασία Χωρο-κειμενικών Συζεύξεων για Δεδομένα Μεγάλης Κλίμακας

(Large-scale Processing of Spatio-textual Joins)

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Αθανάσιος Μουκουβίνας

Επιβλέπων Καθηγητής: Χρήστος Δουλκερίδης

Ιούνιος 2021

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου Χρήστο Δουλκερίδη, ο οποίος με βοηθούσε και καθοδηγούσε κάθε φορά που είχα ανάγκη και ο οποίος μου μετέδωσε γνώση και με συμβούλευε ήδη από την πρώτη στιγμή που επιλέχθηκα για το Μεταπτυχιακό Πρόγραμμα.

Επίσης θα ήθελα να ευχαριστήσω τον φίλο μου Κώστα μιας και ήταν αυτός που με επηρέασε θετικά, μου μετέδωσε την ιδέα να στραφώ σε έναν νέο δρόμο και ειδικότερα σε αυτόν της Επιστήμης των Δεδομένων, με υποστήριξε κατά τη διάρκεια της πορείας μου στο Πρόγραμμα αυτό και πρακτικά με βοήθησε να εξελιχθώ.

Τέλος, θα ήθελα να ευχαριστήσω τη σύντροφό μου Ελένη η οποία ήταν και είναι πάντα δίπλα μου.

Abstract

In our era, it is very popular to search for objects that can be found both between a user-defined distance and having a similarity in their textual features. Big Data are everywhere and are a challenge for the scientific community. It is important to have fast processes that can manage big data sets, so it is of most significance to apply the process regarding the aforementioned search on systems that support distributed processes. Is is also useful to implement efficient methods that reduce the time cost.

For this Thesis, we worked on a centralized environment, we investigated methods of spatio-textual data distribution and processing and we performed a simulation that can be used as a guide for implementation in distributed frameworks. At first, we look into distributing data based on their textual part, but we also worked on distributing data based on their spatial features. For the first method we took advantage of the tokens frequency across the data set. For the second method we partitioned space in zones taking advantage of a data sample and percentiles of their coordinates values Both methods can achieve balanced distribution and each one of them has its preferable advantages.

Περίληψη

Μία από τις πιο δημοφιλείς αναζητήσεις της εποχής μας είναι προσπάθεια ανεύρεσης αντικειμένων τα οποία όχι μόνο βρίσκονται εντός μια επιθυμητής απόστασης, αλλά επιπλέον παρουσιάζουν ομοιότητα σε στοιχεία τα οποία μπορούν να εκφραστούν με τη μορφή κειμένου. Η καθημερινότητά μας χαρακτηρίζεται από τα Μεγάλα Δεδομένα, τα οποία θέτουν μεγάλες προκλήσεις. Προκειμένου να αποκτηθούν γρήγορα τα αποτελέσματα του ερωτήματος μιας τέτοιας αναζήτησης, και να εξαχθούν μέσα από μεγάλα σύνολα δεδομένων, είναι χρήσιμο να τεθεί το ερώτημα αυτό σε συστήματα τα οποία ευνοούν γρήγορη επεξεργασία. Επιπλέον, είναι χρήσιμο να εφαρμοστούν μέθοδοι αποδοτικής κατανομής των δεδομένων προκειμένου να μειωθεί το χρονικό κόστος.

Στην παρούσα εργασία εργαζόμαστε σε κεντρικοποιημένο περιβάλλον και εξετάζονται μέθοδοι κατανομής χωροκειμενικών δεδομένων, προσομοιάζοντας μεθόδους που μπορούν να εφαρμοστούν σε κατανεμημένα συστήματα. Εξετάζεται η κατανομή δεδομένων με βάση το κειμενικό τους μέρος και επίσης εξετάζεται η κατανομή με βάση το χωρικό στίγμα τους. Για την πρώτη μέθοδο εκμεταλλευόμαστε τη συχνότητα που έχουν οι λέξεις μέσα σε ένα σύνολο δεδομένων ενώ για τη δεύτερη μέθοδο χωρίζουμε τον χώρο σε ζώνες εκμεταλλευόμενοι τα ποσοστημόρια των τιμών γεωγραφικού πλάτους τους τα οποία μπορούμε να εξάγουμε από ένα δείγμα των δεδομένων χωρίς μεγάλη επιβάρυνση. Και οι δύο μέθοδοι μπορούν να επιτύχουν εξισορροπημένη κατανομή των δεδομένων με την καθεμία να έχει τα δικά της πλεονεκτήματα και προτιμώμενες εφαρμογές.

Πίνακας περιεχομένων

Πίνακας περιεχομένων σχημάτων

1. Εισαγωγή

Η εποχή μας χαρακτηρίζεται από έναν ωκεανό δεδομένων αλλά και από μία καταιγίδα επεξεργασίας των δεδομένων αυτών. Μία από τις πιο δημοφιλείς λειτουργίες, με πολλές πρακτικές, αφορά τη σύζευξη στοιχείων που βρίσκονται εντός μια επιθυμητής απόστασης και που έχουν συγκεκριμένα επιθυμητά χαρακτηριστικά. Ουσιαστικά, το πρόβλημα που αντιμετωπίζουμε είναι η χωρική και ταυτόχρονα η κειμενική σύζευξη αντικειμένων: Πρακτικά, έχουμε μια συλλογή αντικειμένων τα οποία συνοδεύονται από ένα χαρακτηριστικό τοποθεσίας (συντεταγμένες) αλλά και από ένα χαρακτηριστικό κειμένου (λέξεις) και ο στόχος μας είναι να δημιουργήσουμε ζεύγη αντικειμένων τα οποία παρουσιάζουν μια επιθυμητή ομοιότητα και στα δύο αυτά χαρακτηριστικά. Η εν λόγω διεργασία και η αντιμετώπιση αυτού το προβλήματος, τοποθετείται σε έναν χώρο που αγγίζει την χωρική επεξεργασία, την εξόρυξη δεδομένων και την ανάκτηση πληροφοριών και είναι χρήσιμη στην περίπτωση τοπικοποιημένων προτάσεων. Για παράδειγμα, είναι πολύ συχνή η ανάρτηση κριτικών για καταστήματα ή αξιοθέατα, ή η αναζήτηση ξενοδοχείων ή εστιατορίων συγκεκριμένου τύπου κουζίνας σε συγκεκριμένη περιοχή και επιπροσθέτως είναι μεγάλο το πλήθος των εμπορικών εφαρμογών (Google Maps, TripAdvisor, Booking κ.ά.) οι οποίες προσφέρουν τις αντίστοιχες υπηρεσίες που σχετίζονται με αυτού του είδους τα ερωτήματα. Στην αντιμετώπιση του ανωτέρω προβλήματος έρχεται να προστεθεί ακόμα μία πρόκληση. Οι απαιτήσεις όσον αφορά την αποδοτική απάντηση αυτών των ερωτημάτων, τόσο από άποψη ποιότητας όσο και από άποψη χρόνου, μεγαλώνει καθημερινά. Μια λάθος ή αργή απάντηση είναι μια ανεπιθύμητη έκβαση, τόσο για το σύστημα ή την εφαρμογή που διέπραξε την εν λόγω λειτουργία, όσο και για τον χρήστη που έχει θέσει το ερώτημα. Πλέον απαιτούνται συστήματα που αποτελούνται από περισσότερους και πιο δυνατούς πόρους και πιο έξυπνες αρχιτεκτονικές. Σε αυτές τις προκλήσεις, η λύση έρχεται από τα κατανεμημένα συστήματα επεξεργασίας δεδομένων τα οποία μπορούν να διαμοιράζουν τον φόρτο και να χρησιμοποιούν μια πιο προχωρημένη λογική συγκριτικά με τις παλιές δομές που έκαναν επεξεργασία δεδομένων κεντρικοποιημένα. Με άλλα λόγια η πρόκληση που προστίθεται είναι ο τρόπος με τον οποίο μπορούν να κατανεμηθούν τα δεδομένα σε πολλαπλούς κόμβους επεξεργασίας, με τέτοιο τρόπο που να υπάρχει όσο το δυνατόν μεγαλύτερη μείωση του χρόνου εκτέλεσης και όσο τον δυνατόν καλύτερη εξισορρόπηση του φόρτου των δεδομένων και της επεξεργασίας κατ' επέκταση. Επιπλέον, πρέπει να σημειωθεί ότι το πρόβλημα της σύζευξης, από τη στιγμή που πρέπει να συγκριθούν όλα τα αντικείμενα μιας συλλογής, έχει $O(n^2)$ χρονική πολυπλοκότητα, προσθέτει ακόμα μια πρόκληση για εξεύρεση αποδοτικής λύσης.

Βάση των ανωτέρω, στα επόμενα κεφάλαια της παρούσας Διπλωματικής εργασίας εξετάζονται μέθοδοι κατανομής χωροκειμενικών δεδομένων. Η διαδικασία ανάπτυξης των μεθόδων και των πειραμάτων γίνεται σε κεντρικοποιημένο περιβάλλον με τη γλώσσα Python, κάνοντας μια προσομοίωση μεθόδων που μπορούν να εφαρμοστούν σε κατανεμημένα συστήματα επεξεργασίας δεδομένων. Παίρνοντας υπόψη το κειμενικό μέρος, μπορούμε να αναπτύξουμε μια τεχνική η οποία χειρίζεται το κείμενο που συνοδεύει μια εγγραφή και τη συχνότητα των λέξεων της, ενώ επίσης εξετάζεται η κατανομή με βάση το στίγμα της τοποθεσίας τους χωρίζοντας τον χώρο σε ζώνες εκμεταλλευόμενοι τα ποσοστημόρια που μπορούμε να εξάγουμε από ένα δείγμα των δεδομένων.

1.1 Σκοπός της διπλωματικής εργασίας

Σκοπός της διπλωματικής εργασίας είναι να ερευνηθούν, αναλυθούν και εφαρμοστούν μέθοδοι κατανομής συνόλων δεδομένων με βάση είτε το κειμενικό τους μέρος είτε το χωρικό τους στίγμα, ώστε τελικά να βελτιωθεί η απόδοση ερωτημάτων χωροκειμενικής σύζευξης αντικειμένων. Ο «αφελής» τρόπος επεξεργασίας συζεύξεων χαρακτηρίζεται από σύγκριση όλων των αντικειμένων μιας συλλογής και άρα $O(n^2)$ χρονική πολυπλοκότητα, για αυτό και ο στόχος μας είναι η εξεύρεση περισσότερο αποδοτικών λύσεων, οι οποίες ταυτόχρονα θα έχουν τη δυνατότητα να εφαρμοστούν σε κατανεμημένο περιβάλλον. Οι λύσεις αυτές, θα δίνουν προτεραιότητα στην κατανομή των δεδομένων σε διαχωρισμένα κομμάτια, με όσο το δυνατό καλύτερη εξισορρόπηση, ώστε να υπάρχει δυνατότητα παράλληλης επεξεργασίας των κομματιών αυτών και άρα καλύτερη απόδοση, πρωτίστως από την άποψη του χρόνου εκτέλεσης. Η έρευνα για την παρούσα εργασία θα γίνει σε κεντρικοποιημένο περιβάλλον, προσομοιώνοντας διαδικασίες οι οποίες θα είχαν τη δυνατότητα να υλοποιηθούν και σε κατανεμημένα συστήματα επεξεργασίας δεδομένων.

1.2 Δομή εργασίας

Στο Κεφάλαιο 2 γίνεται μία γενική ανασκόπηση υποβάθρου, όπως επίσης και μια ανασκόπηση δημοσιευμένων μελετών που έχουν διεξαχθεί πάνω στο ζήτημα της κατανομής των δεδομένων και των αλγορίθμων που υλοποιούν κατανεμημένη επεξεργασία δεδομένων.

Στο Κεφάλαιο 3 γίνεται ανάλυση των μεθόδων που αναπτύχθηκαν για την παρούσα εργασία και οι οποίες είναι βασισμένες σε τεχνικές επεξεργασίας δεδομένων που έχουν δημοσιευθεί. Αρχικά αναλύεται μία μέθοδος που διαμοιράζει τα δεδομένα με βάση ιδιότητες των λέξεων που περιέχουν, και ακολούθως αναλύεται μια μέθοδος η οποία κατανέμει τα δεδομένα με βάση το στίγμα της τοποθεσίας με το οποίο είναι συνυφασμένες.

Στο Κεφάλαιο 4 παρουσιάζεται η πειραματική διαδικασία για τις προαναφερθείσες μεθόδους και καταμετρούνται μετρικές όπως η ισορροπία του φόρτου και ο χρόνος εκτέλεσης, αναλόγως και τη μεταβολή σημαντικών παραμέτρων όπως ο αριθμός των partitions και τα κατώφλια ομοιότητας και απόστασης.

Στο Κεφάλαιο 5 ακολουθούν τα συμπεράσματα του έργου της διπλωματικής εργασίας καθώς και προτάσεις για τον τρόπο με τον οποίο μπορεί να συνεχιστεί το εν λόγω έργο.

2. Θεωρητικό υπόβαθρο και σχετικές εργασίες

2.1 Μεγάλα Δεδομένα

Ένα από τα κυριότερα στοιχεία που χαρακτηρίζουν τη σημερινή εποχή είναι το μεγάλο πλήθος των δεδομένων που παράγεται καθημερινά και το οποίο καθορίζει τις περισσότερες πτυχές της εξέλιξης της κοινωνίας. Απόρροια αυτού είναι η γέννηση των Μεγάλων Δεδομένων και ενός τεράστιου αριθμού μεθόδων για τη διαχείριση αυτών.

Ως Μεγάλα Δεδομένα ορίζονται τα σύνολα δεδομένων των οποίων ο τύπος και ο ρυθμός παραγωγής ξεπερνάει τις δυνατότητες των παραδοσιακών σχεσιακών βάσεων δεδομένων όσον αφορά την απόκτηση, διαχείριση και επεξεργασία τους. Κύρια χαρακτηριστικά τους είναι ο μεγάλος όγκος, η υψηλή ταχύτητα και το μεγάλο φάσμα διάφορων τύπων. Τα δεδομένα πλέον προέρχονται από πιο πολύπλοκα συστήματα σε σύγκριση με τα παραδοσιακά δεδομένα μιας και μπορούν να προέρχονται από συστήματα τεχνητής νοημοσύνης, κινητές συσκευές, κοινωνικά δίκτυα, και από όλο τον κόσμο του διαδικτύου. Τα δεδομένα μπορούν πλέον να προέλθουν από αισθητήρες, συσκευές, αρχεία βίντεο και ήχου, δίκτυα, εφαρμογές διάφορων συναλλαγών, διαδικτυακά μέσα ενημέρωσης, και μάλιστα παράγονται πολλές φορές σε πραγματικό χρόνο και σε πολύ μεγάλο εύρος.

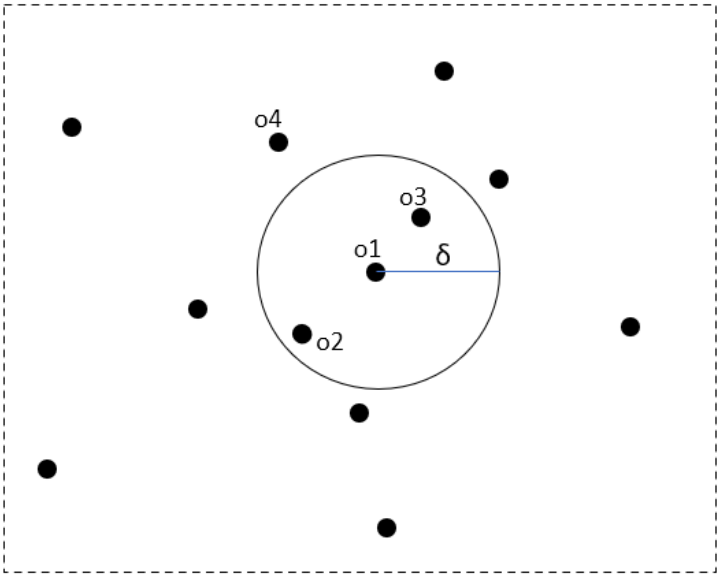
Η Αναλυτική των Μεγάλων Δεδομένων είναι η χρήση προχωρημένων τεχνικών αναλυτικής πάνω σε πολύ μεγάλα και ποικιλόμορφα σύνολα δεδομένων τα οποία συμπεριλαμβάνουν δομημένα, ημι-δομημένα και μη-δομημένα δεδομένα, από διάφορες πηγές και σε διάφορους βαθμούς μεγέθους. Με την Αναλυτική των Μεγάλων Δεδομένων μπορούμε να φτάσουμε σε καλύτερες και πιο γρήγορες αποφάσεις, στη δημιουργία μοντέλων ενισχυμένης ευφυίας και στην πρόβλεψη μελλοντικών εκβάσεων. Το αποτέλεσμα όλων των παραπάνω είναι η ευέλικτη και γρήγορη επεξεργασία και η ανάπτυξη συστημάτων αποθήκευσης που είναι ικανά να διαχειριστούν τον όγκο των δεδομένων που παράγονται τη σημερινή εποχή.

2.2 Χωρικά δεδομένα και δεδομένα κειμένου

Τα χωρικά δεδομένα είναι δεδομένα τα οποία σχετίζονται με γεγονότα ή αντικείμενα τα οποία έχουν σαν ιδιότητα ένα στίγμα μίας τοποθεσίας, όπως φαίνονται στο Σχήμα 1. Μπορούν να είναι είτε στατικά όπως η τοποθεσία ενός κτηρίου, είτε δυναμικά όπως η τροχιά ενός οχήματος. Μπορούν να συμπεριλαμβάνουν τις συντεταγμένες της τοποθεσίας, αλλά μπορούν επίσης να συνοδεύονται από άλλες πληροφορίες που αφορούν το στοιχείο του ενδιαφέροντος.

Επιπλέον, ένας από τους πιο πολυπληθείς τύπους δεδομένων της εποχής μας αφορούν δεδομένα κειμένου. Στην ουσία είναι ένας τεράστιος όγκος δεδομένων που περιβάλλουν όλους τους τομείς της κοινωνίας, από λήμματα εγκυκλοπαιδειών, μέχρι σχόλια σε ιστοσελίδες, εγχειρίδια χρήσεως και κριτικές ταινιών και εστιατορίων όπως φαίνονται στο Σχήμα 2. Φυσικά, μπορούν να συνοδεύονται από χωρικά δεδομένα και ο συνδυασμός τους να προσφέρει ένα μεγάλο βαθμό πληροφοριών σε έναν χρήστη τους. Αξίζει να σημειωθεί

ότι ένα βασικό στάδιο της επεξεργασίας κειμενικών δεδομένων είναι μεταμόρφωση τους σε ένα σύνολο λεξικογραφικών μονάδων. Η διαδικασία αυτή αποκαλείται tokenization και εν λόγω μονάδες αποκαλούνται tokens, τα οποία tokens στην ουσία είναι οι λέξεις ενός κειμένου (ενίοτε μπορούν να είναι ολόκληρες φράσεις). Αυτές οι λέξεις, αναλόγως τις ανάγκες της εκάστοτε διαδικασίας, μπορούν να έχουν περάσει από μια επεξεργασία προκειμένου να παραμείνει μόνο το κομμάτι τους το οποίο συνδέεται με τη ρίζα τους. Για παράδειγμα, η λέξεις «ανθρωπισμός» και «άνθρωπος» μπορούν και οι δύο να επεξεργασθούν ως το token «ανθρωπ».



Σχήμα 1. Παράδειγμα χωρικών δεδομένων

	Review Text	clean_txt
0	Friendly staff, good food and homely environme...	friendly staff good food homely environment 💜💜💜
1	Well...The Food was Good___Intrerior design is...	well...the food good___intrerior design nice e...
2	The man who is foodie like me for him arabian ...	man foodie like arabian master nice place envi...
3	ordered pizza and they were unable to serve th...	order pizza unable serve order set menu take o...
4	This place is too much comfortable & food is d...	place much comfortable food delicious every it...
5	#5star great and very welcoming atmosphere.. p...	great welcome atmosphere pizza great best
6	I check it out like a second home of mine...fe...	check like second home mine...feel frequent re...
7	Arabian Masters interior decoration is unique....	arabian master interior decoration unique.its ...
8	you guys are awesome & I just love your "offer...	guy awesome love offer pizza tasty reasonablep...
9	Well behave Food was testy	well behave food testy

Σχήμα 2. Παράδειγμα δεδομένων κειμένου. Κριτικές εστιατορίων. Αρχικό κείμενο και tokenized κείμενο

2.3 Σχετικές εργασίες

Δοθείσης μιας συλλογής αντικειμένων τα οποία εμπεριέχουν χωρική και κειμενική πληροφορία, μια χωροκειμενική σύζευξη επιστρέφει τα ζευγάρια των αντικειμένων που είναι χωρικά σε κοντινή απόσταση και κειμενικά όμοια [13]. Το ζήτημα έχει διερευνηθεί αναφορικά τόσο με τεχνικές οι οποίες μπορούν να είναι αποδοτικές κατά τον εντοπισμό παρόμοιων στοιχείων, όσο και με τον κατάλληλο διαχωρισμό των συνόλων των στοιχείων με τρόπο που να εξυπηρετεί την κατανεμημένη επεξεργασία των συνόλων αυτών [10][11][12].

2.3.1 Text similarity joins

Οι Rao et al [1] ασχολήθηκαν με την εξέταση στρατηγικών διαχωρισμού των συνόλων αλλά και με τη σύγκριση δύο σημαντικών αλγορίθμων σύζευξης όμοιων συνόλων, των All-Pairs [2] και PPJoin [3] .

Ο αλγόριθμος All-Pairs [2] έχει δύο φάσεις: Στην πρώτη γίνεται δημιουργία ανεστραμμένων ευρετηρίων και στη δεύτερη γίνεται εξεύρεση όμοιων ζευγαριών. Για κάθε token, δημιουργούνται λίστες που περιέχουν την εγγραφή που περιέχει αυτό το token. Για κάθε εγγραφή δημιουργούνται διανύσματα βαρύτητας που διατρέχουν τα tokens της εγγραφής, και το άθροισμα των βαρών αθροίζουν στη μονάδα. Η μέγιστη ομοιότητα που μπορεί να παρουσιάσει μια εγγραφή με με μία άλλη εκφράζεται ως

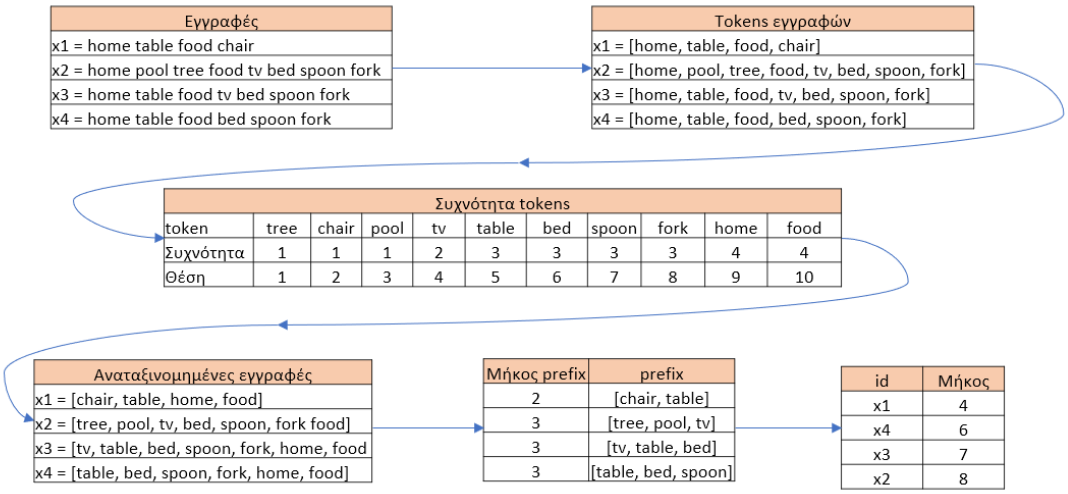
$$\sum_i maxweight_i(R) \cdot x[i]$$

όπου $x[i]$ είναι το βάρος του token t_i και $maxweight_i(R)$ είναι το μέγιστο βάρος του token t_i για όλα τα στοιχεια του συνόλου R. Αν αυτή η τιμή ομοιότητας είναι μικρότερη από το κατώφλι που έχει τεθεί, η εγγραφή x δεν μπορεί να θεωρηθεί παρόμοια με κάποια άλλη.

Ο αλγόριθμος PPJoin [3] κάνει χρήση *length filtering, prefix filtering και position filtering*. Όπως παρατηρούμε στο Σχήμα 3, κατά την προεπεξεργασία γίνεται tokenization των εγγραφών. Έπειτα καταμετρούνται οι συχνότητες των tokens. Τα tokens σε κάθε εγγραφή ταξινομούνται αναλόγως τη γενική συχνότητα των tokens στο σύνολο, δηλαδή από το λιγότερο συχνό προς το πιο συχνό. Υπολογίζεται το μήκος που πρέπει να έχει το prefix σε κάθε εγγραφή με την παρακάτω φόρμουλα

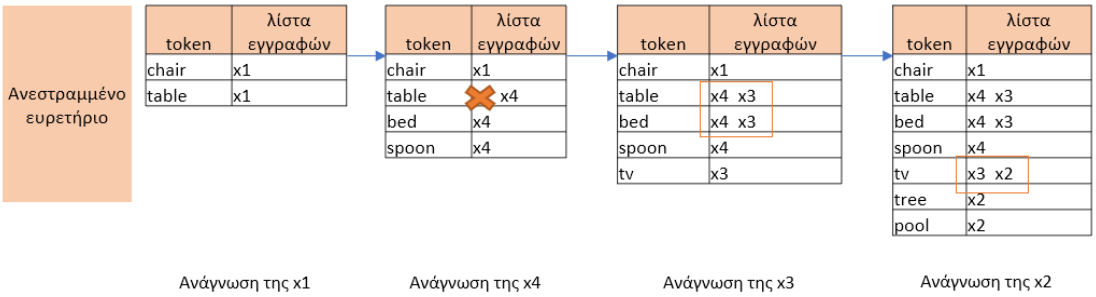
$$|pref(x)| = \lceil (1-t) \cdot |x| \rceil + 1$$

όπου x μια εγγραφή και όπου t ο αριθμός των ελάχιστων κοινών tokens που είναι επιθυμητό να έχουν δύο εγγραφές ώστε να θεωρηθούν όμοιες. Μετά από τον παραπάνω υπολογισμό μένουν ως prefix τα αντίστοιχα tokens σε κάθε εγγραφή. Ακολούθως, οι εγγραφές ταξινομούνται σε αύξουσα σειρά αναλόγως το αρχικό μήκος τους σε tokens.



Σχήμα 3. Η διαδικασία του αλγόριθμου PPJoin

Στη συνέχεια, ο PPJoin δημιουργεί ανεστραμμένα ευρετήρια στα tokens ξεκινώντας από την πιο μικρή εγγραφή. Γίνεται ανάγνωση μιας εγγραφής, το κάθε token του prefix της λειτουργεί σαν κλειδί και στις τιμές του εισάγεται η σχετική εγγραφή. Πηγαίνοντας στην επόμενη εγγραφή, εφαρμόζει το length filter και αν «δει» ότι είναι τόσο μεγαλύτερη από την προηγούμενη ώστε να μην υπάρχει νόημα να συγκριθούν (αναφορικά με την ομοιότητα τους), τα ευρετήρια που είχαν μπει για την πρώτη εγγραφή, διαγράφονται με ασφάλεια και πλέον μένουν τα νέα ευρετήρια. Όταν ένα token αποκτά ευρετήρια για επιπλέον εγγραφές, αυτό σημαίνει ότι αυτές οι εγγραφές θα πρέπει να ελεγχθούν για την ομοιότητά τους.



Σχήμα 4. Ανεστραμμένο ευρετήριο PPJoin

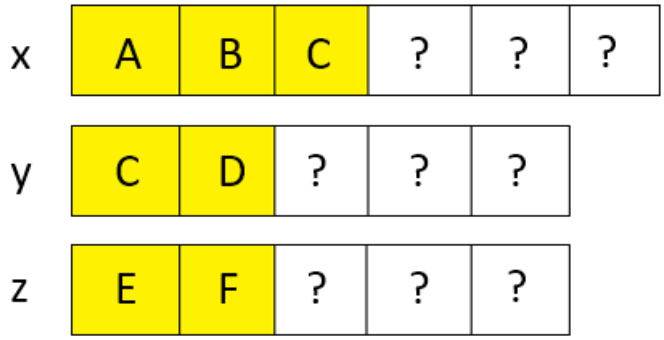
Για παράδειγμα, όπως παρουσιάζεται στο Σχήμα 3, η πρώτη (δηλαδή η πιο μικρή) εγγραφή *x1* έχει μήκος 4 και *pref(x1) = [chair, table]*. Όπως παρουσιάζεται στο Σχήμα 4, το ευρετήριο δημιουργείται με τα δύο αυτά tokens και έκαστο δείχνει την *x1*. Η επόμενη εγγραφή *x4* έχει μήκος 6 και *pref(x4) = [tree, pool, tv]*. Έστω ότι έχει τεθεί κατώφλι ομοιότητας $t = 0.8$. Ο έλεγχος τους length filter μας δείχνει πως

$$|x1| < \lceil t \cdot |x4| \rceil$$

$$4 < \lceil 0.8 \cdot 6 \rceil$$

οπότε η *x1* δεν χρειάζεται να συγκριθεί με την *x4*, ούτε με τις επόμενες εγγραφές οι οποίες έχουν μήκος από 6 και άνω. Επομένως, όταν προστίθενται νέα prefix tokens στον ευρετήριο, όλες οι προσθήκες που αφορούν το *x1*, μπορούν με ασφάλεια να διαγραφούν. Όταν θα γίνει επεξεργασία των επόμενων εγγραφών, *x3* και *x2*, το ανεστραμμένο ευρετήριο δείχνει ότι τα prefix για τα *x4 και x3*, και για τα *x3 και x2*, αλληλεπικαλύπτονται, οπότε οι εγγραφές αυτές χρειάζεται να συγκριθούν για την συνολική ομοιότητά τους.

Έπειτα εφαρμόζεται το prefix filter, το οποίο είναι σημαντικό για περιπτώσεις υψηλών κατωφλιών ομοιότητας. Οδηγούν τα υποψήφια στοιχεία για σύζευξη στο να έχουν μικρό ποσοστό από false positives. Η στρατηγική του *prefix filter* στηρίζεται στο γεγονός ότι παρόμοιες εγγραφές θα πρέπει να έχουν τουλάχιστον ένα κοινό token. Αναλόγως πάντα το κατώφλι ομοιότητας και το μήκος των εγγραφών, θα πρέπει να υπάρχει ταύτιση ακόμα και σε ένα κομμάτι των εγγραφών αυτών. Το συγκεκριμένο filtering βασίζεται στην Αρχή του Περιστερώνα (Pigeonhole Principle). Η εν λόγω Αρχή δηλώνει πως, αν *n* αντικείμενα περιέχονται σε *m* κουτιά, τότε υπάρχει τουλάχιστον ένα κουτί το οποίο δεν περιέχει παραπάνω από *n/m* αντικείμενα. Για παράδειγμα, όπως παρουσιάζεται στο Σχήμα 5, μεταφέροντας αυτή την Αρχή, το prefix filtering λειτουργεί ώς εξής: Σε περίπτωση που ζητείται να υπάρχει ομοιότητα για *t* = 4 tokens μεταξύ δύο εγγραφών, ελέγχουμε τα κοινά tokens των prefix των εγγραφών. Η *x* και η *z* δεν έχουν κανένα κοινό, και βάση των μηκών τους, αυτό σημαίνει ότι σίγουρα αυτές οι δύο εγγραφές δεν έχουν έστω τον ελάχιστο αποδεκτό αριθμό των τεσσάρων κοινών tokens.



Σχήμα 5. Prefix filter

Σαν επιπλέον παράδειγμα, δύο εγγραφές με 10 tokens η καθεμία, και με ζητούμενη ομοιότητα της τάξεως του 0.8 σημαίνει, σύμφωνα με τη συνάρτηση ομοιότητας Jaccard

$$Sim_{Jaccard}(x, y) = \frac{|x \cap y|}{|x \cup y|} = \frac{|x \cap y|}{|x| + |y| - |x \cap y|}$$

ότι θα πρέπει να είναι τέτοια η τομή τους ώστε να συμπίπτουν σε τουλάχιστον 9 tokens.

Δοθέντων δύο συνόλων, r και s, και ένα κατώφλι ταύτισης *t (στην περίπτωση του παραπάνω παραδείγματος, t = 9):* Αν $|r \cap s| \geq t$, τότε υπάρχει τουλάχιστον ένα κοινό token στο *π-prefix* του *r* και *στο π-prefix* του s, όπου $\pi = |r| - t + 1$ (για το r) και $\pi = |s| - t + 1$ (για το s). Για το παραπάνω παράδειγμα, π = 2. Αν σε αυτά τα 2 tokens δεν υπάρχει ταύτιση με τα 2 tokens του prefix άλλης εγγραφής, αυτό το ζευγάρι μπορεί με ασφάλεια να αγνοηθεί. Με άλλα λόγια, θα μας αρκούσε να ελέγξουμε αν συμπίπτουν κομμάτια

αποτελούμενα από 2 tokens. Άμα δεν βρεθεί τέτοια ταύτιση, σημαίνει ότι μπορεί αυτό το ζευγάρι εγγραφών να αγνοηθεί από περαιτέρω έλεγχο.

Αν κριθεί ότι το ζευγάρι χρήζει σύγκρισης, ο PPJoin κάνει έπειτα χρήση του αποκαλούμενου *position filter*. Για να προσεγγίσει τη μέγιστη πιθανή ταύτιση μεταξύ δύο εγγράφων, ο PPJoin εξετάζει τον αριθμό των κοινών prefix tokens όπως επίσης και τη θέση του τελευταίου κοινού prefix token. Με αυτήν την πληροφορία μπορούμε να εξάγουμε τη μέγιστη πιθανή ταύτιση. Η τεχνική του position filter χρησιμοποιήθηκε για την ανάπτυξη της μεθόδου της παρούσας εργασίας και αναλύεται στο Κεφάλαιο 3.

Οι Mann et al [4] διεξήγαγαν πειράματα συγκρίνοντας 7 αλγόριθμους set-similarity join οι οποίοι υιοθετούν μια προσέγγιση επαλήθευσης μέσω φίλτρων. Μέσω μίας τέτοιας προσέγγισης, είναι αρκετό να επιθεωρηθεί μόνο ένα υποσύνολο των δεδομένων που μας απασχολούν, με αποτέλεσμα να έχουμε γρηγορότερους χρόνους εκτέλεσης. Μέσα από αυτή την έρευνα ξεχώρισε η σημασία της τακτικής του prefix filtering και οι αλγόριθμοι All Pairs, PPJoin και GroupJoin [5].

Ο GroupJoin επεκτείνει τον PPJoin και εκμεταλλεύεται το γεγονός ότι διαφορετικά σύνολα δεδομένων μπορούν να έχουν πανομοιότυπα prefixes. Σύνολα με πανομοιότυπα prefixes ομαδοποιούνται και επεξεργάζονται σαν ένα μοναδικό σύνολο κατά τη διάρκεια της δημιουργίας υποψήφιων ζευγαριών, κάτι το οποίο επιτρέπει, σε μεγάλα batches, το κλάδεμα (pruning) υποψηφίων. Τα ομαδοποιημένα υποψήφια ζευγάρια απελευθερώνονται από την ομάδα (group) τους κατά τη φάση του verification.

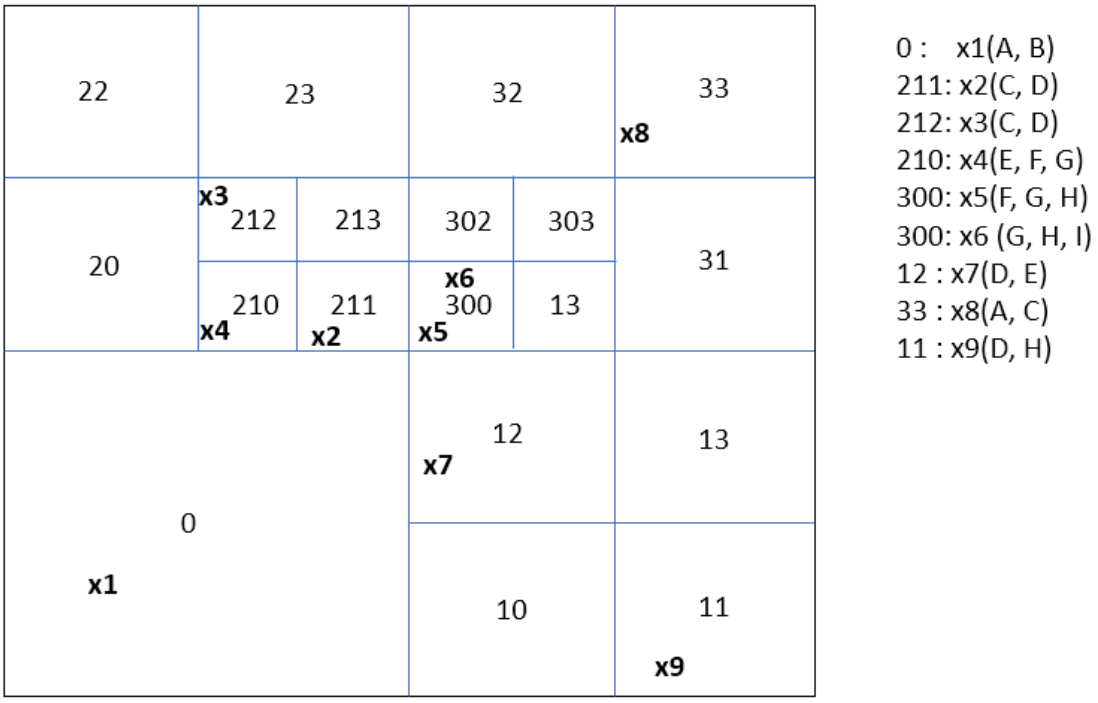
Οι PPJoin και GroupJoin είχαν την καλύτερη μέση απόδοση, ο GroupJoin ήταν ο πιο αξιόπιστος και ο AllPairs ήταν ο καλύτερος όταν έγινε προσπέλαση του μεγαλύτερου αριθμού δεδομένων. Σε γενικές γραμμές οι διαφορές στην απόδοση των 7 αλγορίθμων δεν ήταν μεγάλες. Η φάση της επαλήθευσης ήταν πάντα γρήγορη και τέλος, δεν φάνηκε να έχουν μεγάλη προσφορά τα εξεζητημένα φίλτρα τα οποία κάποιοι αλγόριθμοι, όπως ο AdaptJoin χρησιμοποιούν. Ο AdaptJoin κλαδεύει ένα ζευγάρι (r, s) αν υπάρχουν λιγότερες από e αντιστοιχίες token μεταξύ του e $(\pi r + e - 1)$-prefix του r και του $(\pi s + e - 1)$-prefix του s, όπου π ισοδυναμεί με την τιμή που προαναφέρθηκε στο υποκεφάλαιο του PPJoin. Η τιμή του e εξάγεται από τον AdaptJoin από μία συνάρτηση κόστους [6].

Προς όφελος μίας εξαγωγής συγκεντρωτικών συμπερασμάτων στο [7] γίνεται σύγκριση 10 set similarity join αλγορίθμων οι οποίοι μπορούν να εκτελεστούν σε κατανεμημένο περιβάλλον και έγινε χρήση του framework mapreduce. Κατά τα πειράματα δεν λήφθηκε υπ' οψη η καταμέτρηση του κόστους για τα προ- και μετά- επεξεργαστικά βήματα έτσι ώστε να υπολογιστεί το κεντρικό σημείο των μεθόδων, δηλαδή το ίδιο το similarity join. Μία πρώτη παρατήρηση που εξήχθη από τα πειράματα αυτά ήταν ότι όλοι οι αλγόριθμοι αδυνατούσαν να κάνουν scale για τουλάχιστον ένα σύνολο δεδομένων και είναι ευαίσθητοι σε πολύ μεγάλα σύνολα δεδομένων, σε συχνά στοιχεία, σε χαμηλά κατώφλια ομοιότητας ή σε έναν συνδυασμό όλων των παραπάνω. Επίσης κάποιοι αλγόριθμοι αδυνατούσαν να χειριστούν μικρά σύνολα δεδομένων τα οποία κατά τα άλλα μπορούν να επεξεργαστούν εύκολα σε μη-κατανεμημένο περιβάλλον. Αυτό είναι λογικό μιας και η συνεχής εκκίνηση και παύση εργασιών και η μεταφορά δεδομένων μεταξύ των κόμβων του cluster του MapReduce δημιουργεί υπερφόρτωση. Στις περιπτώσεις μεγάλων συνόλων δεδομένων οι κατανεμημένοι αλγόριθμοι είναι προτιμότεροι.

2.3.2 Spatio-textual joins

Στη μελέτη των Rao et al [1], αναφορικά με τις στρατηγικές διαχωρισμού, εξετάστηκαν οι εξής δύο ιδέες:

Κατά την πρώτη, προηγείται ο διαχωρισμός του χώρου, είτε με πλέγμα (grid) είτε με quadtrees [14] (Σχήμα 6) και ακολούθως οι χωρικές περιοχές (partitions) διατρέχονται από έναν αλγόριθμο ομοιότητας ο οποίος δημιουργεί ανεστραμμένα ευρετήρια στις περιοχές και ο οποίος εν τέλει εντοπίζει όμοια στοιχεία τα οποία ταυτόχρονα βρίσκονται εντός επιθυμητής απόστασης (Local approach).



Σχήμα 6. Quadtree

Κατά τη δεύτερη, χωρίζεται ο χώρος, έπειτα ακολουθεί η δημιουργία ανεστραμμένων ευρετηρίων (Σχήμα 7) στο σύνολο των δεδομένων και ακολούθως ξεχωρίζεται χωρικά κάθε μία από τις καταχωρήσεις που δημιουργήθηκαν για κάθε token κειμένου (Global approach).

A	<x1, 0>	<x8, 33>		
B	<x1, 0>			
C	<x2, 211>	<x3, 212>	<x8, 33>	
D	<x2, 211>	<x3, 212>	<x7, 12>	<x9, 11>
E	<x4, 210>	<x7, 12>		
F	<x4, 210>	<x5, 300>		
G	<x4, 210>	<x5, 300>		
H	<x5, 300>	<x6, 300>	<x9, 11>	
I	<x6, 300>			

Σχήμα 7. Posting λίστες της global approach

Ένα επιπλέον πεδίο το οποίο εξετάστηκε στην ίδια εργασία ήταν η συμπεριφορά των τεχνικών αναλόγως του περιβάλλοντος στο οποίο αναπτύχθηκαν, δηλαδή σε μεμονωμένο υπολογιστικό πυρήνα ή σε ομάδα πυρήνων. Το συμπέρασμα στο οποίο κατέληξε η εν λόγω έρευνα ενισχύει την άποψη ότι στη σύγχρονη εποχή των διαμοιρασμένων συστημάτων και των πολλαπλών υπολογιστικών πυρήνων, είναι απαραίτητη η εξισορρόπηση του φόρτου αλλά και η πλήρης εκμετάλλευση των παραπάνω περιβαλλόντων.

Η δημιουργία ευρετηρίων βρέθηκε να είναι πιο γρήγορη κατά την global προσέγγιση αλλά η συνολική απόδοση ήταν γρηγορότερη στις περιπτώσεις της local προσέγγισης. Επίσης παρατηρήθηκε ότι υψηλότερα κατώφλια ομοιότητας οδηγούν σε μικρότερους χρόνους επεξεργασίας, κάτι το οποίο συμβαίνει και όσο μικραίνει το κατώφλι απόστασης. Με το τελευταίο σχετίζεται και η παρατήρηση ότι ο συνολικός χρόνος εκτέλεσης μια διαδικασίας μειώνεται όταν μειώνεται και ο μέσος αριθμός στοιχείων σε κάθε κόμβο του χώρου. Σε multi-threaded περιβάλλοντα, το partitioning της global προσέγγισης ήταν γρηγορότερο της local και επιπλέον τα εν λόγω περιβάλλοντα ευνοούν την επεξεργασία skewed συνόλων δεδομένων. Ακόμα, οι τεχνική του quadtree φαίνεται να προσφέρει ταχύτερους χρόνους συγκριτικά με τα grids για unbalanced δεδομένα. Επιπροσθέτως, παρατηρήθηκε ότι ο συνδυασμός PPJoin και της συνάρτησης ομοιότητας Jaccard ήταν ο πιο γρήγορος.

3. Σχεδιασμός και εφαρμογή τεχνικών διαχείρισης χωρο-κειμενικών δεδομένων

3.1 Προσέγγιση προβλήματος

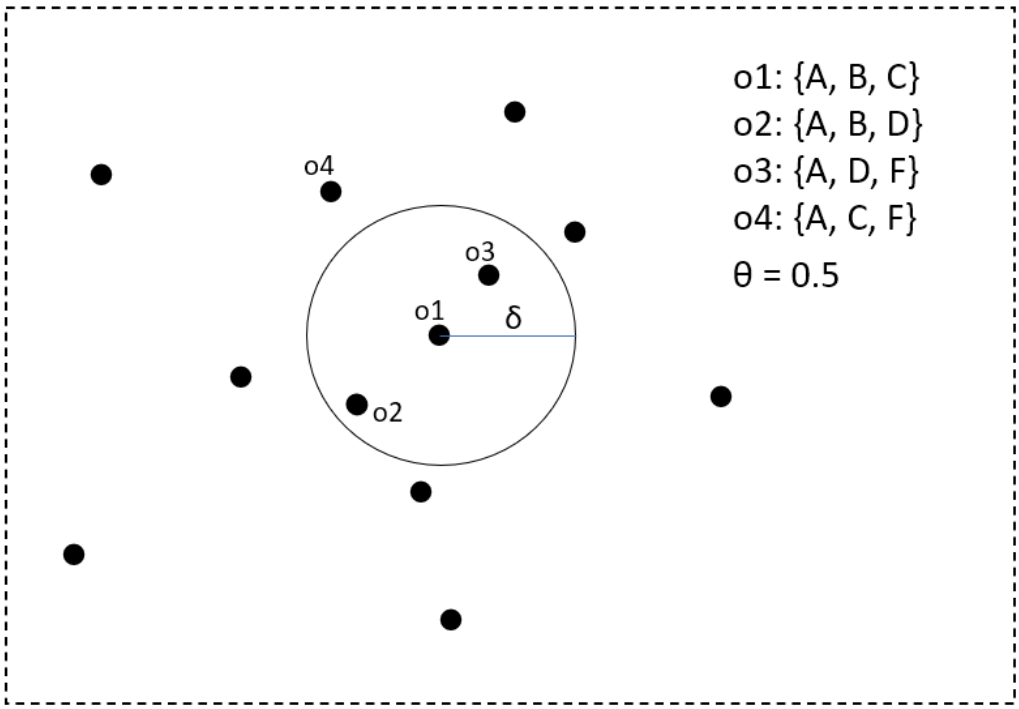
Κύριο θέμα της παρούσας διπλωματικής εργασίας είναι η σύζευξη χωροκειμενικών εγγραφών οι οποίες προέρχονται από διαφορετικά σύνολα δεδομένων και οι οποίες παρουσιάζουν ομοιότητα στο κειμενικό τους μέρος και επιπλέον, βάση του στίγματος των συντεταγμένων τους, βρίσκονται εντός μιας συγκεκριμένης απόστασης. Δοθέντων δύο συνόλων δεδομένων *S* και *T*, μίας συνάρτησης ομοιότητας *SIM*, ένα κατώφλι ομοιότητας θ, μίας συνάρτησης απόστασης *DIS* και ένα κατώφλι απόστασης δ, το πρόβλημα που θέτεται είναι το να βρεθούν όλα τα ζεύγη $(s, t) \in (S, T)$ για το οποία θα ισχύει $SIM(s, t) \geq \theta$ και $DIS(s, t) \leq \delta$.

Για τα πειράματα της εργασίας έγινε χρήση της συνάρτησης ομοιότητας Jaccard

$$Sim_{Jaccard}(x, y) = \frac{|x \cap y|}{|x \cup y|} = \frac{|x \cap y|}{|x| + |y| - |x \cap y|}$$

και της συνάρτησης η οποία μετράει την Ευκλείδεια απόσταση.

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}$$



Σχήμα 8. Δεδομενα κειμένου στον χώρο

Στο Σχήμα 8, υπάρχουν σαν δεδομένα η ακτίνα δ του κύκλου, το κατώφλι ομοιότητας θ = 0.5 και τα κειμενικά μέρη των εγγραφών o1, o2, o3, o4. Με βάση την συνάρτηση ομοιότητας Jaccard, η εγγραφή o1 είναι ικανοποιητικά όμοια με την εγγραφή o4 :

$$\frac{2}{3+3-2} = 0.5$$

αλλά η σύζευξή τους απορρίπτεται από τη διαδικασία μας μιας και απέχουν απόσταση μεγαλύτερη του δ. Αντιστοίχως, οι εγγραφές o1 και o3 έχουν κοντινά στίγματα, αλλά η σύζευξη τους απορρίπτεται μιας και η ομοιότητά τους είναι μικρότερη από το κατώφλι θ:

$$\frac{1}{3+3-1} = 0.2$$

Επιπλέον, εξ ίσου σημαντικό ζήτημα το οποίο και εξετάστηκε, είναι ο διαχωρισμός των εγγραφών σε κομμάτια (partitions) ώστε να προσομοιωθεί η αντιστοίχισή τους στους κόμβους ενός κατανεμημένου συστήματος επεξεργασίας, έτσι ώστε να μπορεί να διευκολυνθεί η επεξεργασία τους από άποψη κόστους χρόνου. Προκειμένου να επιτευχθεί βελτίωση της απόδοσης, δηλαδή εν προκειμένω ελαχιστοποίηση του μέγιστου χρόνου επεξεργασίας, τα partitions πρέπει να έχουν όσο το δυνατόν λιγότερες εγγραφές, να υπάρχει εξισορρόπηση του φόρτου μεταξύ τους και να μπορούν να επεξεργαστούν παράλληλα έτσι ώστε να υπάρχει η δυνατότητα να γίνει χρήση ενός κατανεμημένου συστήματος το οποίο στην τελική φάση θα ενοποιεί τα αποτελέσματα του παραλληλισμού και θα δίνει το τελικό αποτέλεσμα των όμοιων ζευγαριών. Ιδανικά, κάθε partition θα πρέπει έχει όσο το δυνατόν λιγότερα διπλότυπα, αν και αυτή η πρόκληση αγγίζει την spatial-first προσέγγιση που θα παρουσιάσουμε, μιας και η text-first χρησιμοποιεί ένα κάθετο κομμάτιασμα των κειμενικών εγγραφών με αποτέλεσμα να μην υπάρχει ο κίνδυνος διπλότυπων, αλλά ένας σχετικός κίνδυνος να είναι partitions που δεν έχει μειωθεί αρκετά ο φόρτος τους σε σύγκριση με τον φόρτο του πλήρους συνόλου δεδομένων.

Για τις απαιτήσεις της παρούσας εργασίας, μελετήθηκαν και προσομοιώθηκαν τεχνικές partitioning, οι οποίες βασίζονται σε δύο διαφορετικές λογικές. Η πρώτη είναι μία text-first λογική και αφορά τον διαχωρισμό των εγγραφών με βάση το κειμενικό τους μέρος. Η δεύτερη είναι μία spatial-first λογική και αφορά των διαχωρισμό τους με βάση τα στίγματα της τοποθεσίας τους.

3.2 Text-first

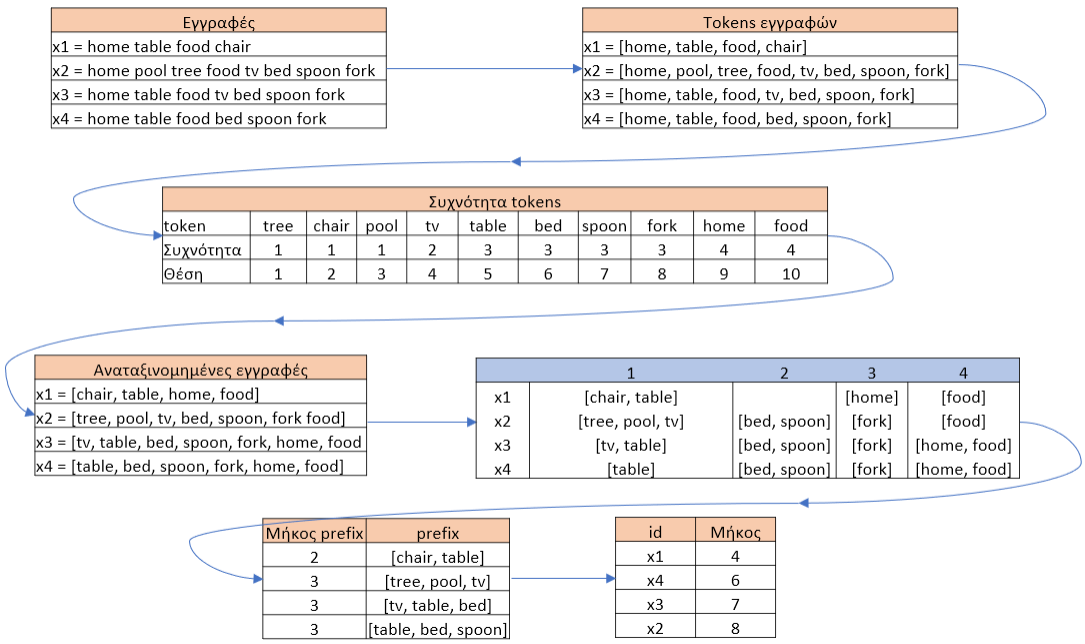
Για την συγκεκριμένη λογική εφαρμόστηκαν τεχνικές οι οποίες έχουν παρουσιαστεί στα [3] και [8] στο οποίο αναπτύχθηκε ο αλγόριθμος FSJoin. Στην παρούσα εργασία ακολουθήθηκε ένας συνδυασμός επεξεργασίας του αρχικού συνόλου δεδομένων και φίλτρων ώστε αρχικά να γίνεται ομαδοποίηση των εγγραφών και έπειτα να γίνεται «κλάδεμα» (pruning) ζευγαριών τα οποία δεν θα καλύπτουν τα κατώφλια ομοιότητας και απόστασης. Ακολουθώντας τις τεχνικές που προτείνονται για τον FSJoin [8], η μέθοδος του partitioning που ερευνήθηκε ακολουθεί τον διαχωρισμό των κειμενικών μερών των εγγραφών με έναν κάθετο τρόπο. Τα κειμενικά μέρη κομματιάζονται και τα tokens για κάθε εγγραφή μοιράζονται σε έναν αριθμό κομματιών (segments). Όλες οι εγγραφές έχουν

τον ίδιο αριθμό segments, και αυτά τα segments ακολουθούν τα ίδια όρια-pivots τα οποία βασίζονται σε μια ανάλυση που γίνεται πάνω στη συχνότητα των tokens στο σύνολο δεδομένων. Τα segments, με κριτήριο την αύξουσα σειρά τους, αποτελούν στο σύνολο τους ένα fragment (Σχήμα 9). Όλα τα πρώτα segments των εγγραφών, συνθέτουν το πρώτο fragment. Το fragment αυτό αποτελεί μία ομάδα εγγραφών αυξημένης ομοιότητας για τις οποίες γίνεται η επιλογή να επεξεργαστούν στον ίδιο κόμβο ενός κατανεμημένου συστήματος επεξεργασίας.

Τα segments κάθε εγγραφής καθορίζονται από κάποια σταθερά pivots που ορίζουν που θα «κοπεί» κάθε εγγραφή. Με αυτή την τεχνική, σε κάθε fragment (και άρα partition) μπορεί να υπάρχει ένα επιπλέον φίλτρο στην αρχή της διαδικασίας το οποίο θα αποκλείει ζευγάρια παίρνοντας υπ' όψη τα μήκη των segments και τα κοινά tokens τους. Πρόκειται για μια text-first προσέγγιση κατά την οποία δεν δημιουργούνται ανεπιθύμητα duplicates μέσα στον ίδιο κόμβο. Επίσης εκμεταλλεύεται την αύξουσα σειρά (βάση της συχνότητας) των tokens ώστε να βρεθούν τα σταθερά pivots για το χώρισμα των segments, και με αυτό τον τρόπο, κάποιες εγγραφές έχουν κενά segments στα fragments, οπότε δεν φορτώνουν με δεδομένα το partition. Επιπλέον αυτή η τεχνική εύρεσης των pivots συμβάλλει σε μια εξισορρόπηση φόρτου.

	1	2	3	4
x1	[chair, table]		[home]	[food]
x2	[tree, pool, tv]	[bed, spoon]	[fork]	[food]
x3	[tv, table]	[bed, spoon]	[fork]	[home, food]
x4	[table]	[bed, spoon]	[fork]	[home, food]

Σχήμα 9. Τέσσερα fragments αποτελούμενα από τα segments τεσσάρων εγγραφών



Σχήμα 10. Τεχνική partitioning των κειμενικών μερών των εγγραφών

Πιο συγκεκριμένα, όπως παρουσιάζεται στο Σχήμα 10, για τη συγκεκριμένη μέθοδο, αρχικά συλλέγουμε όλα τα tokens του συνόλου δεδομένων. Ακολούθως, βρίσκουμε τη συχνότητα με την οποία εμφανίζεται κάθε token συνολικά στο σύνολο δεδομένων. Με βάση αυτή τη συχνότητα, ταξινομούμε τα tokens σε αύξουσα σειρά. Έπειτα εκμεταλλευόμαστε αυτή τη σειρά και μεταμορφώνουμε τις εγγραφές ταξινομώντας τα tokens τους με βάση αυτήν.

Για παράδειγμα, αν οι συχνότητες *{token: συχνότητα}* που προκύψουν για τα tokens ενός συνόλου δεδομένων, φέρουν το αποτέλεσμα:

{every: 124}, {food: 342}, {day: 526}, {love: 689}

Τότε η εγγραφή **[love, food, every, day]** θα μετατραπεί σε **[every, food, day, love]**

Στο σχήμα 11 παρουσιάζεται ένα σύνολο με τη σχετική ταξινόμηση των κειμενικών μερών.

```
print(sorted_by_freq)
```

```
{'B&B': 1, 'golf': 1, 'smoking_porch': 1, 'cribs_surcharge': 4, 'beach': 5, 'bo
utique': 6, 'family': 7, 'hostel': 12, 'internet_in public areas only': 26, 'fi
tness_facilities_nearby': 29, 'extended stay': 38, 'conference center': 59, 'co
ndo': 69, '': 103, 'airport': 171, 'resort': 183, 'spa_services_beauty service
s': 210, 'pets_deposit required': 237, 'in_room_massage': 319, 'parking_secur
e': 349, 'parking_garage': 362, 'spa_services_steam room': 421, 'business': 45
3, 'parking_surcharge': 671, 'internet_surcharge': 683, 'no-step showers': 759,
'secretarial_services': 779, 'wedding_services': 818, 'currency_exchange': 828,
'motel': 831, 'braille signage': 874, 'accessible bathroom': 954, 'equipment fo
r the deaf': 1034, 'handicapped parking': 1052, 'express_check_out': 1569, 'pet
s_fee': 1587, 'express_check_in': 1681, 'spa_services_massage': 2189, 'non_smok
ing_rooms': 2630, 'elevator': 3357, 'event_catering': 3646, 'smoking': 3875, 's
pa_services_sauna': 4091, 'complimentary_newspapers': 4139, 'cribs': 4221, 'par
king_valet': 4315, 'spa_services_whirlpool': 4539, 'internet_wired': 4624, 'twe
ntyfour_hour_front_desk': 4720, 'concierge': 4768, 'parking_self': 4860, 'banqu
et_facilities': 4947, 'wheelchair accessible': 5038, 'spa_services_jacuzzi': 56
79, 'meeting_rooms': 6685, 'laundry_service': 6864, 'spa_services': 7092, 'pet
s': 7194, 'internet_wireless': 7434, 'complimentary_breakfast': 7882, 'cable_t
v': 7890, 'internet_free': 7978, 'fitness_facilities': 7990, 'internet': 8678,
'parking_free': 8841, 'air_conditioning': 9106, 'parking': 9164}
```

Σχήμα 11. Tokens συνόλου δεδομένων σε αύξουσα σειρά βάση της συχνότητάς τους

Έπειτα αναζητούμε τα pivots τα οποία θα καθορίζουν σε πιο σημείο θα «κόβεται» κάθε εγγραφή σε segment. Τρεις τρόποι pivoting είναι:

• Τυχαία επιλογή: Κάθε token έχει την ίδια πιθανότητα να επιλεχθεί ως pivot και έπειτα επιλέγονται τόσα όσος και ο αριθμός των partitions που θέλουμε. Αυτή η μέθοδος όμως δεν μπορεί να εξασφαλίσει εξισορρόπηση του φόρτου.

• Ίσα διαστήματα: Χωρίζουμε το group των ταξινομημένων tokens σε ίσα μέρη. Με αυτόν τρόπο εξασφαλίζουμε ότι κάθε segment θα αποτελείται από τα ίδια tokens σε κάθε εγγραφή. Όμως, λόγω των διαφορών στις συχνότητες, παρότι η εξισορρόπηση του φόρτου θα ήταν βελτιωμένη συγκριτικά με τον προαναφερθέν τρόπο, και πάλι δεν θα ήταν ικανοποιητική

• Ίσες αθροιστικές συχνότητες (Even-TF): Αυτός είναι ο τρόπος που ακολουθήθηκε στην παρούσα εργασία μιας και μπορεί να έρθει περισσότερο ικανοποιητικά στο στόχο του, όχι μόνο να αποτελείται κάθε segment από τα ίδια tokens σε κάθε εγγραφή, αλλά επίσης να αποτελείται και κάθε fragment από ίσο αριθμό tokens σε σύγκριση με τα άλλα fragments.

Με βάση την παραπάνω λογική, προκειμένου να υπάρχει μια ισορροπία στα fragments, εκμεταλλευόμαστε ξανά τη αύξουσα σειρά με την οποία έχουμε ταξινομήσει τα tokens του συνόλου μας. Αρχικά, βρίσκουμε το άθροισμα των εμφανίσεων των tokens σε όλο το σύνολο δεδομένων. Έπειτα, θέτουμε κάποια pivots τα οποία σχετίζονται με τον αριθμό των partitions στα οποία θέλουμε να χωρίσουμε το σύνολο δεδομένων. Ξεκινώντας από την αρχή, κάνουμε καταμέτρηση της τιμής των συχνοτήτων και κάθε φορά κάνουμε παύση όταν η καταμέτρηση ξεπεράσει την τιμή:

$$W \cdot \frac{1}{P} \cdot i$$

όπου:

- W: Άθροισμα των παρουσιών των tokens στο σύνολο δεδομένων
- P: Αριθμός των partition που θέλουμε να χωρίσουμε τα δεδομένα
- i: ο αριθμός του partition. Για παράδειγμα, για το πρώτο partition, $i = 1$, για το δεύτερο. $i = 2$ κ.ο.κ.

Ο αριθμός των pivots θα είναι ίσος με P - 1 .

```
print(pivots_list)
['cribs', 'meeting_rooms', 'cable_tv']
```

Σχήμα 12. Pivots για διαδικασία κατανομής σε 4 partitions

Για παράδειγμα, αν θέλουμε 4 partitions και η αθροισμένη συχνότητα μέχρι-ένα-token-πριν το *«meeting rooms»* αγγίζει το 50% του αριθμού των tokens του συνόλου δεδομένων, τότε κρατάμε το *«meeting rooms»* σαν το pivot όπου σε μέγιστο βαθμό σε αυτό θα τελειώνει το 2ο segment σε κάθε εγγραφή, με βάση την ταξινόμηση που έχουμε δημιουργήσει στα tokens (Σχήμα 12). Με βάση αυτά τα pivots, χωρίζουμε κάθε εγγραφή σε segments. Επομένως, αν για παράδειγμα μια εγγραφή δεν έχει στη σύνθεσή της κανένα token του οποίου η συχνότητα να βρίσκεται ανάμεσα στη συχνότητα δύο pivots, τότε αυτό το segment θα είναι κενό. Αυτό φυσικά βοηθάει στο να μειωθεί το μέγεθος ενός fragment και άρα και ο φόρτος του κόμβου που θα επεξεργαστεί το fragment αυτό. Όλων των εγγραφών τα πρώτα segments θα περιέχονται στο πρώτο fragment (και άρα partition) και παρομοίως το ίδιο θα συμβει με τα επόμενα fragments.

Από την παραπάνω διαδικασία εξάγουμε τη σημαντική πληροφορία του μεγέθους (μήκους) για κάθε segment. Η επόμενη πληροφορία που πρέπει να συλλεχθεί, αναφέρεται συχνά στη βιβλιογραφία ως το «αριστερό» και «δεξιό» μέρος (Left part, Right part) του κειμενικού κομματιού.

• Left part: ο αριθμός των tokens που βρίσκονται σε προηγούμενη θέση από τη θέση του πρώτου token ενός segment. Αυτό σημαίνει ότι το Left part του πρώτου segment είναι πάντα μηδενικό.

• Right part: ο αριθμός των tokens που βρίσκονται σε επόμενη θέση από τη θέση του τελευταίου token ενός segment. Αυτό σημαίνει ότι το Right part του τελευταίου segment είναι πάντα μηδενικό.

Επιπλέον, προκειμένου να γίνουν στη συνέχεια οι απαραίτητες συγκρίσεις, κομβικό προπαρασκευαστικό κομμάτι αυτής της μεθόδου είναι η χρήση της τεχνικής του *prefix filtering.*

Εξάγουμε το πρόθεμα (prefix) κάθε εγγραφής με βάση τον τύπο:

$$|pref(x)| = \lceil (1-t) \cdot |x| \rceil + 1$$

Όπου *t* το κατώφλι ομοιότητας.

Έπειτα ταξινομούμε τις εγγραφές μέσα στο σύνολο δεδομένων με βάση το μήκος τους, σε αύξουσα σειρά. Η πιο μικρή εγγραφή πλέον βρίσκεται στην αρχή του συνόλου δεδομένων. Αυτό θα μας βοηθήσει στην εφαρμογή ενός *length filtering*.

```
['Courtyard-Los Angeles Torrance',
 33.822559999999996,
 -118.332474,
 'B',
 8,
 ['non_smoking_rooms',
  'internet_wireless',
  'internet_free',
  'fitness_facilities',
  'internet',
  'parking_free',
  'air_conditioning',
  'parking'],
 ['non_smoking_rooms', 'internet_wireless', 'internet_free'],
 [],
 0,
 8,
 ['non_smoking_rooms'],
 0,
 7,
 [],
 0,
 7,
 [],
 0,
 7,
 [],
 0,
 7,
 [],
 0,
 7,
 [],
 0,
 7,
 ['internet_wireless'],
 1,
 6,
 [],
 1,
 6,
 ['internet_free', 'fitness_facilities'],
 2,
 4,
 ['internet', 'parking_free'],
 4,
 2,
 ['air_conditioning', 'parking'],
 6,
 0]
```

Σχήμα 13. Τελική μορφή μιας εγγραφής.

Στο Σχήμα 13 παρουσιάζεται ενδεικτικά η τελική μορφή μιας εγγραφής πριν κατανεμηθεί. Περιέχει αναγνωριστικό *("Courtyard-Los Angeles Torarnce"),* συντεταγμένες *(33.82, -118.33),* ένδειξη του συνόλου όπου άνηκε *("B"),* μήκος κειμενικού μέρους *(8)*, κειμενικό μέρος *([non_smoking_rooms, ... , parking])*, prefix *([non_smoking_rooms, ... internet_free])*, segments, segment left part και segment right part. Για παράδειγμα, το πρώτο segment είναι κενό και το μήκος του left part του είναι 0 ενώ το μήκος του right part του είναι 8.

Μετά το στάδιο της προεπεξεργασίας, το σύνολο δεδομένων διαχωρίζεται σε fragments τα οποία μπορούν να επεξεργαστούν ξεχωριστά σε διαφορετικούς κόμβους του συστήματος.

Έπειτα για κάθε fragment τρέχει παράλληλα ο αλγόριθμος σύζευξης, ο οποίος για εγγραφές που προέρχονται από δύο σύνολα *S* και *T*, συγκρίνει το αντίστοιχο segment της εγγραφής *s* έναντι αυτού της *t*. Ο αλγόριθμός συγκρίνει κάθε εγγραφή με τις επόμενες της, κάνοντας κατά σειρά συγκεκριμένους ελέγχους. Φυσικά ο αρχικός είναι:

• Ελέγχει αν προέρχονται από διαφορετικά σύνολα δεδομένων.

Έπειτα, προκειμένου να μειωθεί ο χρόνος επεξεργασίας, κρίνεται σημαντικό το να μπουν σε εφαρμογή συγκεκριμένα φίλτρα τα οποία θα συγκρίνουν ένα μικρό κομμάτι του κειμενικού μέρους της εγγραφής. Αν από αυτή τη σύγκριση δεν προκύψει ότι ένα ζευγάρι είναι πιθανό να είναι όμοια, τότε απορρίπτεται. Με αυτόν τον τρόπο, στον τελικό έλεγχο ομοιότητας μπαίνουν λιγότερα ζευγάρια.

• Φίλτρο μήκους segment (segment length filtering): Σε αυτό το στάδιο, Ελέγχεται αν ισχύει

$$\min\left(\left|Seg_s^i\right|, \left|Seg_t^i\right|\right) < \frac{\theta}{1+\theta} \cdot (|s| + |t|) - \min(|s^h|, |t^h|) - \min\left(|s^t, r^t|\right)$$

όπου

- $\left|Seg_s^i\right|, \left|Seg_t^i\right|$: μήκη segment s, t
- θ: κατώφλι ομοιότητας,
- $|s|$, $|t|$: μήκος εγγραφών s, t
- $|s^h|, |t^h|$: μήκη των left part των segments για s, t αντιστοίχως
- $|s^t, r^t|$: μήκη των right part των segments για s, t αντιστοίχως

Σε περίπτωση που ισχύει, αυτό σημαίνει ότι το ζευγάρι δεν είναι ικανοποιητικά όμοιο και επομένως απορρίπτεται. Διαφορετικά μπορεί να ελεγχθεί με επόμενο φίλτρο.

• Έλεγχος ομοιότητας prefix: Για αυτό το φίλτρο καταμετράται πόσα όμοια tokens έχουν οι δύο εγγραφές στα prefix τους. Αν δεν έχουν όμοια tokens, δεν υπάρχει περίπτωση ομοιότητας και άρα το ζευγάρι απορρίπτεται. Διαφορετικά μπορεί να ελεγχθεί με επόμενο φίλτρο

• Σε περίπτωση που βρέθηκαν όμοια tokens, η διαδικασία προχωράει σε Position filtering με τον εξής τρόπο [3][9]:

- κρατάει το τελευταίο (αυτό που βρίσκεται πιο «δεξιά» / που έχει τη μεγαλύτερη γενική συχνότητα) κοινό τους token,
- υπολογίζει για καθεμία από τις δύο συγκρινόμενες εγγραφές πόσα tokens έχουν μετά το token αυτό (ένα «right part» – "rp"),
- κρατάει τον μικρότερο αριθμό από τους δύο (το μικρότερο rp length), και σε αυτόν προσθέτει τον αριθμό των κοινών prefix tokens αυτών των εγγραφών.

Με βάση τα παραπάνω, ελέγχεται αν

$$Sim_{jaccard}(x, y) \geq t \Leftrightarrow Overlap(x, y) \geq \left\lceil \frac{t}{1+t} \cdot (|x| + |y|) \right\rceil = a$$

Όπου

- $MaxOverlap(x, y) = |lp(x) \cap lp(y)| + \min\left(|rp(x)|, |rp(y)|\right)$
- η τομή στην ουσία εκφράζει τον αριθμό των κοινών prefix tokens.

- $|rp(x)|$ και $|rp(y)|$ είναι τα μήκη των *rp* που προαναφέρθηκαν
- *t*, το κατώφλι ομοιότητας

Η διαδικασία επομένως ελέγχει αν το *MaxOverlap* είναι μεγαλύτερο απο το *minimum* αναγκαίο overlap.

• Αν δεν είναι μεγαλύτερο, το ζευγάρι απορρίπτεται. Διαφορετικά υπάρχει πιθανότητα ομοιότητας και οπότε γίνεται verified, δηλαδή γίνεται υπολογισμός για την ομοιότητα των δυο συγκρινόμενων εγγραφών με τη συνάρτηση ομοιότητας Jaccard.

• Αν το ζευγάρι είναι όμοιο, η διαδικασία συνεχίζει στον τελευταίο έλεγχο όπου ελέγχεται αν οι δύο συγκρινόμενες εγγραφές είναι εντός του κατωφλιού απόστασης που έχουμε θέσει. Αν ναι, αυτό το ζευγάρι αποθηκεύεται (Σχήμα 14).

```
[['Meadowbrook Bed and Breakfast', 'Red Brick Inn'],
 ['Charles Brush Lodge', 'The Ida Jean Bed and Breakfast'],
```

Σχήμα 14. Pairing εγγραφών

Στο Σχήμα 15 παρουσιάζεται ο ψευδοκώδικας για την text-first προσέγγιση.

```
1:  input w(loc, text, t, d)
2:  function text-first:
3:  count tokens frequency (F)
4:  sort textual part of records by F
5:  calculate pivots
6:  extract prefix
7:  calculate segments
8:  sort r by length
9:  for i ∈ r do
10:         for j ∈ r+1 do
11:             if segment length filter is passed then
12:                 if prefix filter is passed then
13:                     if position filter is passed then
14:                         if dist(i,j) ≤ d then
15:                             achieve join of (i,j)
16:                         end if
17:                     end if
18:                 end if
19:             end if
20:         end for
21: end for
22: end function
```

Σχήμα 15. Ψευδοκώδικας text-first

3.3 Spatial-First

Η μέθοδος αυτή δίνει προτεραιότητα στον διαχωρισμό των δεδομένων με βάση το στίγμα τους στο χώρο. Στοχεύει στο να αποκτήσουν οι εγγραφές μια ένδειξη ότι ανήκουν σε μία ζώνη-γειτονιά, που οριοθετείται από κάποια όρια. Με αυτόν τον τρόπο, όσες εγγραφές έχουν την ίδια ένδειξη, θα ανήκουν στο ίδιο partition το οποίο θα επεξεργαστεί αυτόνομα. Λόγω της ιδιαιτερότητας της διάσπασης, πρέπει να υπάρξει μία διαδικασία κατά την οποία, για σημεία τα οποία βρίσκονται κοντά στα όρια των ζωνών και τα οποία ενδέχεται να πρέπει να ενωθούν με σημεία άλλης ζώνης, να δημιουργηθεί ένα αντίγραφο αυτών στην αντίστοιχη ζώνη. Αυτή τη διαδικασία πρέπει να γίνει με τέτοιο τρόπο ώστε οι διαδικασίες που θα ακολουθηθούν να αποφεύγουν, όσο είναι δυνατό, να αυξήσουν το χρονικό κόστος.

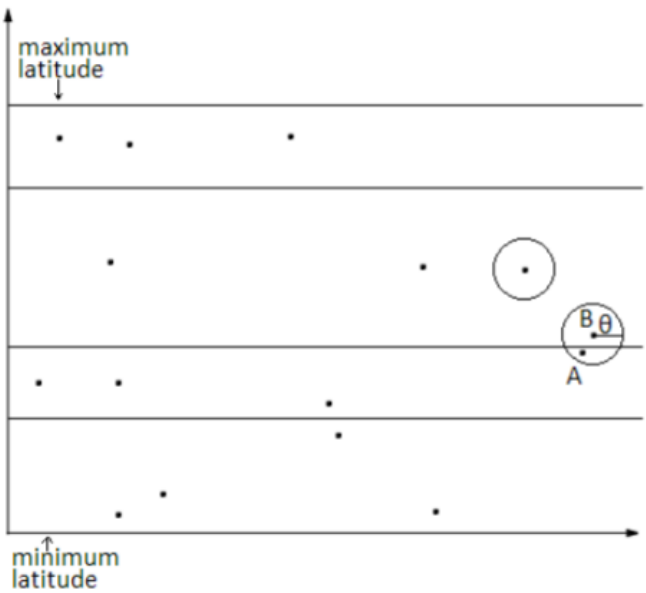
Η διαδικασία που αναπτύχθηκε είχε ως μέλημα συγκεκριμένες σχεδιαστικές αρχές:

• Να εξασφαλίζει την ομαδοποίηση κοντινών σημείων-γειτόνων μιας ζώνης, διαχωρίζοντας τον χώρο στον οποίο βρίσκονται.

• Την εξασφάλιση ότι σημεία τα οποία βρίσκονται κοντά σε όρια ζωνών, αν η δοθείσα απόσταση θ το επιβάλλει, να μπορούν να αντιγραφούν αποκτώντας ένδειξη της ζώνης στην οποία είναι πιθανό να βρουν σημεία για σύζευξη. Αυτό θα πρέπει να λειτουργεί ακόμα και για περιπτώσεις όπου ένα σημείο θα πρέπει να αντιγραφεί, όχι μόνο σε επόμενη, αλλά και σε μεθεπόμενη ζώνη.

• Να προσφέρει ικανοποιητική ισορροπία του φόρτου (load balancing) για οποιονδήποτε τύπο κατανομής των δεδομένων.

Για τα παραπάνω σχεδιάστηκε μία λύση η οποία χωρίζει τον χώρο σε οριζόντιες ζώνες με βάση το γεωγραφικό πλάτος (latitude) και δίνοντας σε κάθε εγγραφή, ως επιπλέον χαρακτηριστικά, τα όρια των ζωνών (Σχήμα 16).



Σχήμα 16. Το latitude σημείου που ανήκει στο σετ B μειωμένο κάτα ϑ, υποδηλώνει ότι πρέπει να αντιγραφεί και στην προηγούμενη ζώνη. Στο παράδειγμα του σχήματος υπάρχει ένα σημείο που ανήκει στο A και θα έπρεπε να συζευχθεί μαζί του.

Η μέθοδος η οποία αναπτύχθηκε, έχει τη δυνατότητα να διασπά τα δεδομένα σε όσα partitions ζητηθούν από τον χρήστη. Ο αριθμός των συνόλων δεδομένων που θέλουμε να συζευχθούν είναι δύο. Η αρχική κίνηση της προεπεξεργασίας είναι πρόσθεση σε κάθε εγγραφή μιας διακριτής ένδειξης αναλόγως το σύνολο από το οποίο προέρχεται. Ακολούθως τα σύνολα ενώνονται σε ένα σετ και πλέον πρέπει να βρεθούν τα όρια της ζώνης στην οποία θα ανήκουν.

Ένας συνηθισμένος τρόπος διαχωρισμού στοιχείων σε ζώνες, και μάλιστα ταξινομημένα, είναι να γίνει αρχικά ταξινόμηση των εγγραφών με βάση το στοιχείο που μας ενδιαφέρει (για παράδειγμα, στην περίπτωση μας, το latitude), έπειτα σειριακή αρίθμηση κάθε εγγραφής με έναν αριθμό, διαίρεση του αριθμού αυτού με τον αριθμό τον κομματιών στα οποία είναι επιθυμητό να χωριστούν και τελικά απομόνωση του ακέραιου αριθμού του αποτελέσματος. Όμως στην περίπτωση των κατανεμημένων συστημάτων επεξεργασίας, πάνω στη λογική των οποίων έγινε η προσομοίωση της εργασίας, μία τέτοια επεξεργασία θα ήταν ακριβή μιας και η ταξινόμηση θα ανάγκαζε τα δεδομένα σε μετακίνηση και αναδιανομή μεταξύ των κόμβων. Αυτή η μετακίνηση των στοιχείων είναι μία από τις πιο επιβαρυντικές διαδικασίες για τα κατανεμημένα περιβάλλοντα και για αυτό δεν αναπτύχθηκε για την παρούσα εργασία παρότι οδηγεί με μαθηματική ακρίβεια σε άψογο load balancing. Στην παρούσα εργασία επιλέχθηκε να αναπτυχτθεί μια διαδικασία που αφορά την εξεύρεση ποσοστημορίων με βάση το latitude σε ένα δείγμα των αρχικών δεδομένων. Για να γίνει αυτό χωρίς επιβάρυνση, κατά σειρά

• Έγινε δειγματοληψία του συνόλου δεδομένων.

• Έγινε χρήση της μεθόδου Quantile της Python. Πρόκειται για μία αξιόπιστη μέθοδο η οποία επιστρέφει προσεγγιστικά ποσοστημόρια για τα αριθμητικά δεδομένα μας. Ο χρήστης μπορεί να επιλέξει το βαθμό προσέγγισης ώστε κατά περίπτωση να θυσιάσει ακρίβεια προς χρόνο και μνήμη ή αντιστρόφως. Για την ανάπτυξη της παρούσας εργασίας έγινε, πάνω σε μικρό δείγμα, εξεύρεση ποσοστημορίων με βάση τα latitude των εγγραφών. Αυτά αποθηκεύτηκαν σε μία λίστα υποδηλώνοντας τα όρια που χωρίζουν τα σημεία στον χώρο, σε ζώνες προσεγγιστικά ίσου μεγέθους. Η μέθοδος που αναπτύχθηκε δύναται να βρίσκει όρια αναλόγως τον αριθμό των ζητούμενων partitions.

Στη συνέχεια της διαδικασίας

• Το latitude κάθε εγγραφής συγκρίθηκε με τα όρια και στην εγγραφή προστέθηκε η απαραίτητη ένδειξη ζώνης

Στην επόμενη φάση

• Έγινε έλεγχος στα σημεία. Σε περίπτωση που το latitude τους, με πρόσθεση ή αφαίρεση της απόστασης θ, κάνει κάποια σημεία πιθανά να ενωθούν με σημεία που ανήκουν σε άλλη ζώνη, αυτό μας αναγκάζει να πρέπει να τα αντιγράψουμε δίνοντας τους πλέον την ένδειξη της ζώνης στην οποία είναι πιθανό να βρουν σημείο για σύζευξη.

Επιλέγουμε ένα από τα δύο σετ των οποίων τα σημεία θα ελεγχθούν ώστε αν χρειαστεί να αντιγραφούν. Η διαδικασία που ακολουθήθηκε είναι η εξής:

Αρχικά, λαμβάνουμε υπ' όψιν ότι το θ δίνεται σε χιλιόμετρα ενώ το latitude σε μοίρες. Μία μοίρα latitude κυμαίνεται από 110,57 χιλιόμετρα (στον Ισημερινό) εως 111,69 χιλιόμετρα (στους Πόλους). Αυτό σημαίνει ότι: 1 μοίρα = (απόσταση σε km) / ~111. Έστω

x η τιμή «(απόσταση σε km)/~111». Προκειμένου να μη χάσουμε συζεύξεις, πρέπει να είμαστε σίγουροι ότι το x θα είναι ικανοποιητικά μεγάλο ώστε να «εισβάλει» σε διπλανή ζώνη. Για αυτό, στη θέση του ~111 έγινε χρήση της τιμής 110,57. Προστέθηκε σε κάθε εγγραφή ένα χαρακτηριστικό με την τιμή latitude.εγγραφής + x. Για τα σημεία που προέρχονται από ένα από τα δύο σύνολα (στα πειράματα επιλέχθηκε το «B»), αν αυτό το ποσό είναι μεγαλύτερο από την τιμή ορίου μιας επόμενης ζώνης, τότε η εγγραφή θα πρέπει να αντιγραφεί σε αυτή τη ζώνη. Αντιστοίχως προστέθηκε σε κάθε εγγραφή ένα χαρακτηριστικό με την τιμή latitude.εγγραφής - x. Αν αυτό το ποσό είναι μικρότερο από την τιμή ορίου μιας προηγούμενης ζώνης, τότε η εγγραφή θα πρέπει να αντιγραφεί σε αυτή τη ζώνη.

	dataset_id	lat	lon	name	p_id	sum_minus	sum_plus	tokens
0	B	3.806059	6.250485	I820Z	4.0	3.715616	3.896502	[4167, 4736, 1202, 2633, 2178, 4773, 2783, 932...
1	A	1.010354	6.997385	OVIM1	1.0	0.919911	1.100797	[2750, 275, 1662, 366, 367, 1518, 3017, 325, 3...
2	A	5.491205	4.833988	GNNKY	6.0	5.400762	5.581648	[4378, 2328, 2953, 1708, 3716, 1328, 1555, 422...
3	A	0.137706	14.597785	K4BB5	0.0	0.047263	0.228149	[2938, 726, 4437, 249, 214, 2830, 4066, 261, 4...
4	A	0.965856	9.494830	7IRT2	1.0	0.875413	1.056299	[1242, 2784, 4043, 2396, 791, 4989, 3938, 3134...
...	...	...	...	...	...	...	...	...
9138	B	9.193570	15.048598	6O53C	11.0	9.103127	9.284013	[4773, 0, 3996, 3552, 1998, 1332, 1776, 1887, ...
9403	B	9.199517	2.556667	QKMVG	11.0	9.109074	9.289960	[4995, 4440, 3996, 4884, 1221, 0, 444, 222, 27...
9558	B	9.180356	2.818006	HJD9Y	11.0	9.089913	9.270799	[888, 4551, 3108, 3774, 4107, 555, 4884, 2997,...
9592	B	9.239268	21.752461	AHFIZ	11.0	9.148825	9.329711	[2886, 4440, 333, 4551, 4773, 3774, 3108, 0, 5...
9764	B	9.227491	0.012510	DB9EW	11.0	9.137048	9.317934	[3552, 3441, 1332, 4440, 666, 4107, 4329, 0, 2...

11562 rows × 8 columns

Σχήμα 17. Εγγραφές μετασχηματισμένες ώστε να μπορέσουν, να χρειαστεί, να γίνουν duplicate

Για να γίνουν οι απαραίτητες αντιγραφές ο αλγόριθμος εκτελεί μια διαδικασία με τις απαραίτητες συνθήκες ώστε να δημιουργηθούν οι επιπλέον εγγραφές που θα αντιγραφούν σε κάθε ζώνη. Στο τέλος, οι αρχικές και οι επιπλέον εγραφές ενώνονται. Προσομοιώνοντας ένα κατανεμημένο σύστημα, πλέον μπορεί να γίνει partitioning με βάση την ένδειξη της ζώνης (p_id στο Σχήμα 17) και έτσι θα μπορούν οι executors να επεξεργαστούν ανεξάρτητα και παράλληλα το ερώτημα για κάθε ζώνη.

Στο τελικό στάδιο της μεθόδου, είναι δυνατόν σε κάθε partition να συγκριθούν οι εγγραφές με βάση την ομοιότητά τους και την απόστασή τους και να προκύψουν οι τελικές συζεύξεις.

Στο Σχήμα 18 παρουσιάζεται ο ψευδοκώδικας για την spatial-first προσέγγιση.

```
1:  input w(loc, text, t, d)
2:  function spatial-first:
3:  extract sample
4:  calculate latitude percentiles
5:  create duplicates
6:  for i ∈ r do
7:          for j ∈ r+1 do
8:                  if dist(i,j) ≤ d then
9:                          if sim(i,j) ≥ t then
10:                                 achieve join of (i,j)
11:                         end if
12:                 end if
13:         end for
14: end for
15: end function
```

Σχήμα 18. Ψευδοκώδικας spatial-first

4. Πειραματική διαδικασία

Κατά την πειραματική διαδικασία έγιναν δοκιμές και εφαρμόστηκε η text-first και η spatial-first τεχνική που αναλύθηκαν στο Κεφάλαιο 3. Πραγματοποιήθηκε προσομοίωση μίας κατανεμημένης διαδικασίας η οποία είχε ως αποτέλεσμα τον διαχωρισμό (partitioning) συνόλων δεδομένων. Επιπλέον, στο πλαίσιο των δοκιμών, δοκιμάστηκαν διαφορετικές τιμές σημαντικών μεταβλητών που καθορίζουν τη λειτουργία τέτοιου είδους διαδικασιών.

Για τις ανάγκες των πειραμάτων, έγιναν δοκιμές τόσο σε συνθετικό όσο και σε πραγματικό σύνολο δεδομένων.

Τα πειράματα πραγματοποιήθηκαν σε κεντρικοποιημένο περιβάλλον, σε υπολογιστή με επεξεργαστή Intel Core i7 1,8 GHz 4 πυρήνων, 8GB RAM, OS Windows 10. Έγινε χρήση της γλώσσας Python.

Οι μετρήσεις που καταγράφηκαν αφορούσαν

- Χρόνο εκτέλεσης
- Εξισορρόπηση φόρτου, δηλαδή κατά πόσο ήταν ισορροπημένος ο αριθμός των εγγραφών που ανατέθηκε σε κάθε partition. Μετρήθηκε πόσες εγγραφές ανατέθηκαν σε κάθε διαμέριση
- Ποσοστό φόρτου σε κάθε partition σε σύγκριση με το μέγεθος του αρχικού συνόλου δεδομένων
- Διπλότυπες εγγραφές στο ίδιο partition
- Pruning, δηλαδή πόσα ζευγάρια φιλτραρίστηκαν από τους πρώτους ελέγχους των μεθόδων, με αποτέλεσμα να μην συμπεριληφθούν στον τελικό έλεγχο ομοιότητας.

4.1 Σύνολα δεδομένων

4.1.1 Συνθετικό σύνολο δεδομένων

Με κώδικα Python, δημιουργήθηκε ένα συνθετικό σύνολο δεδομένων 10.000 εγγραφών με τα παρακάτω attributes:

- ID
- Κείμενο
- Latitude
- Longitude

Κάθε εγγραφή είχε τη μορφή λίστας με περιεχόμενο τα παραπάνω στοιχεία. Το στοιχείο του κειμένου είχε τη μορφή λίστας της οποίας τα στοιχεία ήταν τα tokens. Όλες οι εγγραφές ανήκαν σε μια υπερ-λίστα. Το κειμενικό μέρος είχε μέγεθος από 15 έως 25 tokens και τα

πιθανά tokens τα οποία μπορούσαν να συνθέσουν ένα κείμενο, ήταν 5000. Για το χωρικό στίγμα, το longitude μπορούσε να πάρει τιμές από 0 έως 25 και το latitude από 0 έως 10 και η επιλογή έγινε τυχαία με ομοιόμορφο τρόπο (Σχήμα 19).

```
['2CYJ5', [4925, 4995, 3670, 1244, 1785, 903, 3180, 3779, 277, 2165, 3271, 4709, 4292, 4225, 3363, 4980, 3723, 177, 1196, 3264,
1303, 1386, 722], 9.503376, 4.718227, 'B']
```

Σχήμα 19. Εγγραφή από το συνθετικό σύνολο δεδομένων. Έχει προστεθεί χαρακτηριστικό/ένδειξη συνόλου προέλευσης

4.1.2 Πραγματικό σύνολο δεδομένων

Στη δεύτερη φάση των πειραμάτων, οι παραπάνω μέθοδοι δοκιμάστηκαν σε ένα σύνολο πραγματικών δεδομένων (Σχήμα 20). Έγινε χρήση δεδομένων ξενοδοχείων, τα οποία συνοδεύονται από το γεωγραφικό τους στίγμα, αλλά και από ένα κειμενικό κομμάτι όπου αναφέρονται οι υπηρεσίες που προσφέρουν στους φιλοξενούμενούς τους. Χρησιμοποιήθηκε δείγμα 10.000 εγγραφών, μοιρασμένο σε δύο σύνολα, με τα παρακάτω attributes:

- dataset_id
- lat (latitude)
- lon (longitude)
- name (αναγνωριστικό εγγραφής)
- services_tokens (κειμενικό μέρος)

	dataset_id	lat	lon	name	p_id	services_tokens
23599	A	42.306734	-71.382794	Courtyard By Marriott Natick/Framingham	10.0	[express_check_in, express_check_out, laundry_...
432	A	32.760983	-117.166320	Best Western Seven Seas	3.0	[laundry_service, cable_tv, currency_exchange,...
12929	B	28.236221	-82.728102	Quality Inn and Suites Conference Center	1.0	[air_conditioning, internet, internet_free, in...
23618	A	41.648736	-70.988360	Comfort Inn North Dartmouth	9.0	[laundry_service, cable_tv, concierge, elevato...
23489	B	42.308700	-71.386100	Boston Framingham Red Roof	10.0	[cable_tv, currency_exchange, twentyfour_hour_...

Σχήμα 20. Εγγραφές πραγματικού συνόλου δεδομένων, σε στάδιο όπου έχει προστεθεί η ένδειξη ζώνης/partition p_id

4.2 Αποτελέσματα πειραμάτων

4.2.1 Text-first με συνθετικό σύνολο δεδομένων

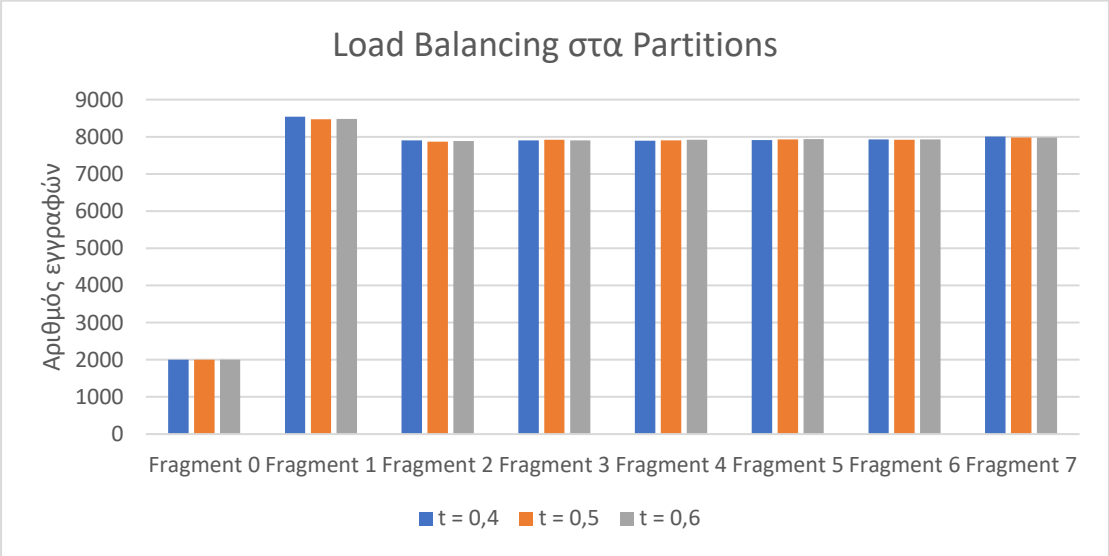
Η πρώτη μέθοδος για την οποία έγιναν δοκιμές αφορούσε την text-first προσέγγιση και την κατανομή δεδομένων με βάση το κειμενικό τους μέρος. Οι δοκιμές, όπως αναφέρθηκε, έγιναν σε 10.000 εγγραφές και υπήρξαν οι εξής παραλλαγές σε σημαντικές μεταβλητές της διαδικασίας:

t	P	d
0,4	8	10
0,5	8	10
0,6	8	10
0,5	4	10
0,5	8	10
0,5	12	10

Όπου:

- t: κατώφλι ομοιότητας
- P: αριθμός Partitions
- d: κατώφλι απόστασης. Δεν μεταβλήθηκε στα text-first πειράματα μιας και το επίκεντρο αυτών δεν είναι οι χωρικές αποστάσεις.

Το πρώτο βήμα από το οποίο μπορούν να εξαχθούν συμπεράσματα, αφορά τον φόρτο ο οποίος κατανέμεται σε κάθε partition

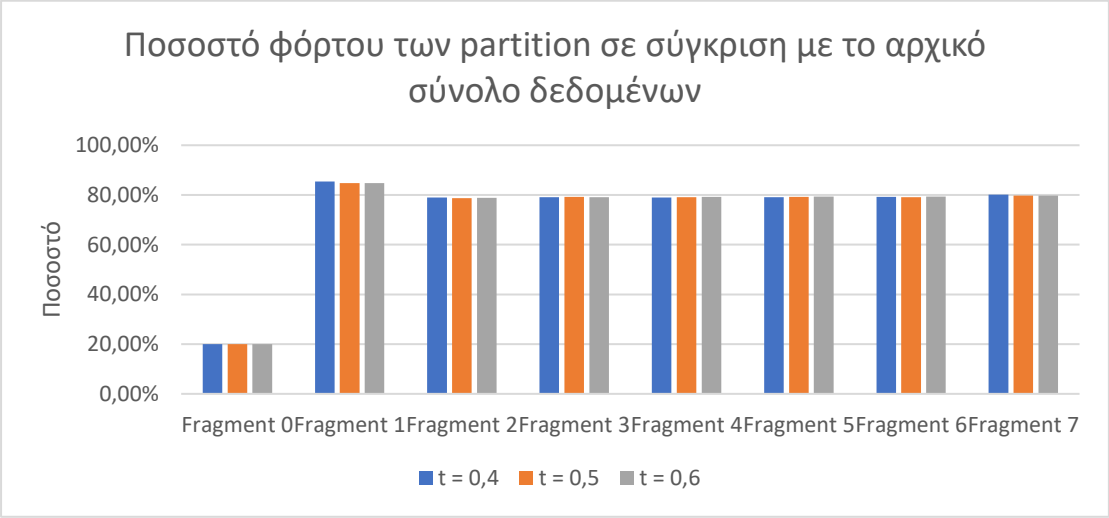


Σχήμα 21. Load Balancing

Στο Σχήμα 21 παρουσιάζεται η κατανομή του πλήθους των εγγραφών ανά fragment. Το πρώτο συμπέρασμα που μπορεί να εξαχθεί είναι ότι η μέθοδος εξασφαλίζει μία ισορροπημένη κατανομή εγγραφών στη συντριπτική πλειοψηφία των partitions. Το συγκεκριμένο αποτέλεσμα είναι απόρροια της μεθόδου Ίσων Συχνοτήτων στην επιλογή των token pivots που χωρίζουν τα segments, όπως περιεγράφηκε στο Κεφάλαιο 3. Οι αλλαγές στην τιμή του κατωφλιού ομοιότητας δεν επηρεάζουν την κατανομή, ο οποίος ακολουθεί μια σταθερή τιμή. Παρ' όλα αυτά, στα πρώτο fragment συγκεντρώνονται αρκετά λιγότερες εγγραφές το οποίο μεταφράζεται στο ότι υπάρχουν πολλά κενά πρώτα segments. Αυτό εξηγείται ως εξής:

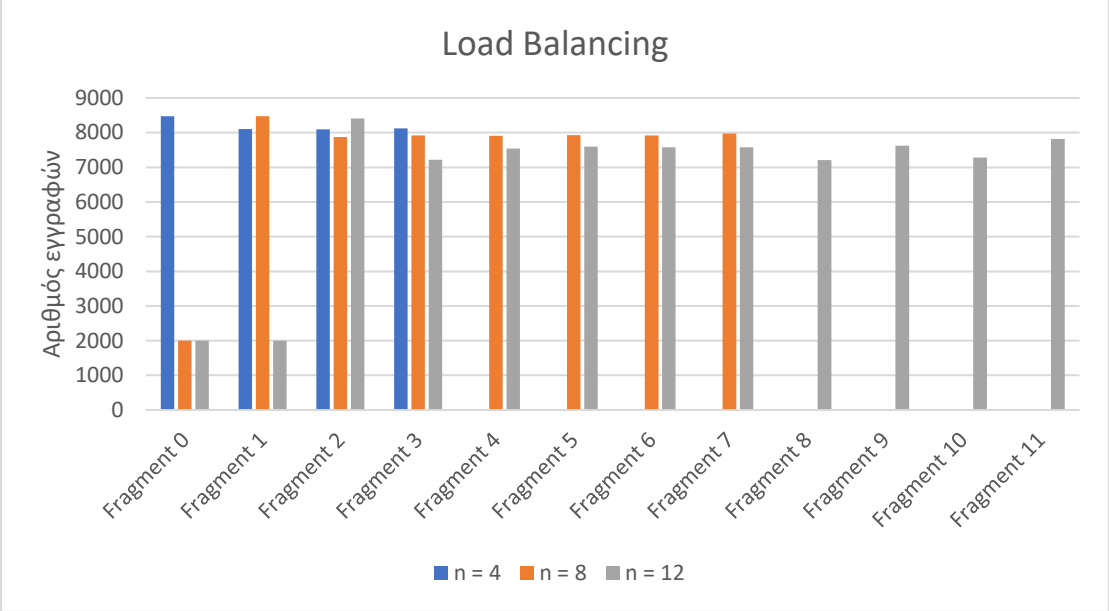
Όπως αναφέρθηκε στο Κεφάλαιο 3, στον πυρήνα των αλγορίθμων που επεξεργάζονται κείμενο, βρίσκεται η ταξινόμηση των tokens κάθε έγγραφης σε μια αύξουσα σειρά η οποία βασίζεται στην γενική αύξουσα σειρά των tokens βάση της συχνότητάς τους σε ολόκληρο το σύνολο δεδομένων. Με άλλα λόγια, στα πρώτα segments ανήκουν πάντα τα σπάνια tokens με αποτέλεσμα να είναι πολύ συχνό το φαινόμενο να είναι κενά μιας και αυξάνονται οι πιθανότητες για μια εγγραφή να μην περιέχει τα εν λόγω tokens. Όπως γίνεται αντιληπτό, αυτό μπορεί να επηρεαστεί από την ιδιομορφία ενός συνόλου δεδομένων καθώς κάποιο σύνολο που αποτελείται από ισορροπημένης συχνότητας tokens, θα έχει και λιγότερες διακρίσεις σε σπάνια και συχνά tokens.

Αξίζει να επαναληφθεί πως, όπως περιεγράφηκε στο Κεφάλαιο 3, και όπως φαίνεται στο Σχήμα 9, κάθε fragment αποτελείται από segments εγγραφών. Το πρώτο fragment έχει τα πρώτα segments των εγγραφών. Αν μία εγγραφή έχει κενό πρώτο segment, τότε το πρώτο fragment δεν θα περιέχει το segment αυτής της εγγραφής. Όσο περισσότερα κενά segments παρουσιάζονται, τόσο λιγότερο φόρτο θα έχουν τα αντίστοιχα fragments. Σε κάθε fragment (και άρα partition), μπορεί να υπάρχει μόνο μία φορά το segment μίας εγγραφής, αρα μπορεί να υπάρχει μόνο μία φορά και ολόκληρη η εγγραφή που συνοδεύει το segment. Οπότε μέσα στο ίδιο partition, δεν υπάρχει πάνω από μία φορά η ίδια εγγραφή. Από την άλλη όμως, αυτό από το οποίο υποφέρει αυτή η τεχνική, είναι το γεγονός ότι μία εγγραφή μπορεί να έχει segments τα οποία θα μπουν και σε άλλα fragments, άρα αυτή η εγγραφή θα υπάρχει σε παραπάνω από ένα partitions. Σε σχέση με τα ανωτέρω, είναι δυνατό και χρήσιμο να υπολογιστεί ο ποσοστιαίος φόρτος των εγγραφών σε κάθε partition σε σύγκριση με το μέγεθος του αρχικού συνόλου δεδομένων.

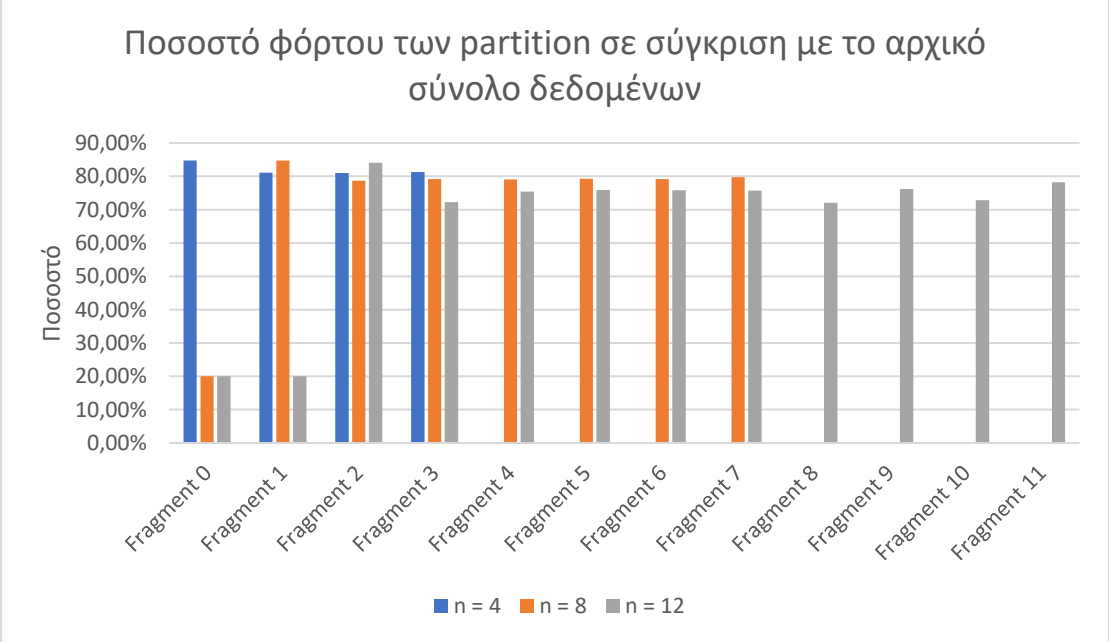


Σχήμα 22. Ποσοστό φόρτου

Στο Σχήμα 22 παρατηρούμε ότι ο φόρτος είναι της τάξης του 80% για τα περισσότερα partitions, με το πρώτο partition να διαφέρει για τους λόγους που προαναφέρθηκαν. Αντιστοίχως με τα παραπάνω μπορούμε να ερευνήσουμε πως συμπεριφέρεται η διαδικασία όταν αλλάζει ο επιθυμητός αριθμός partitions.



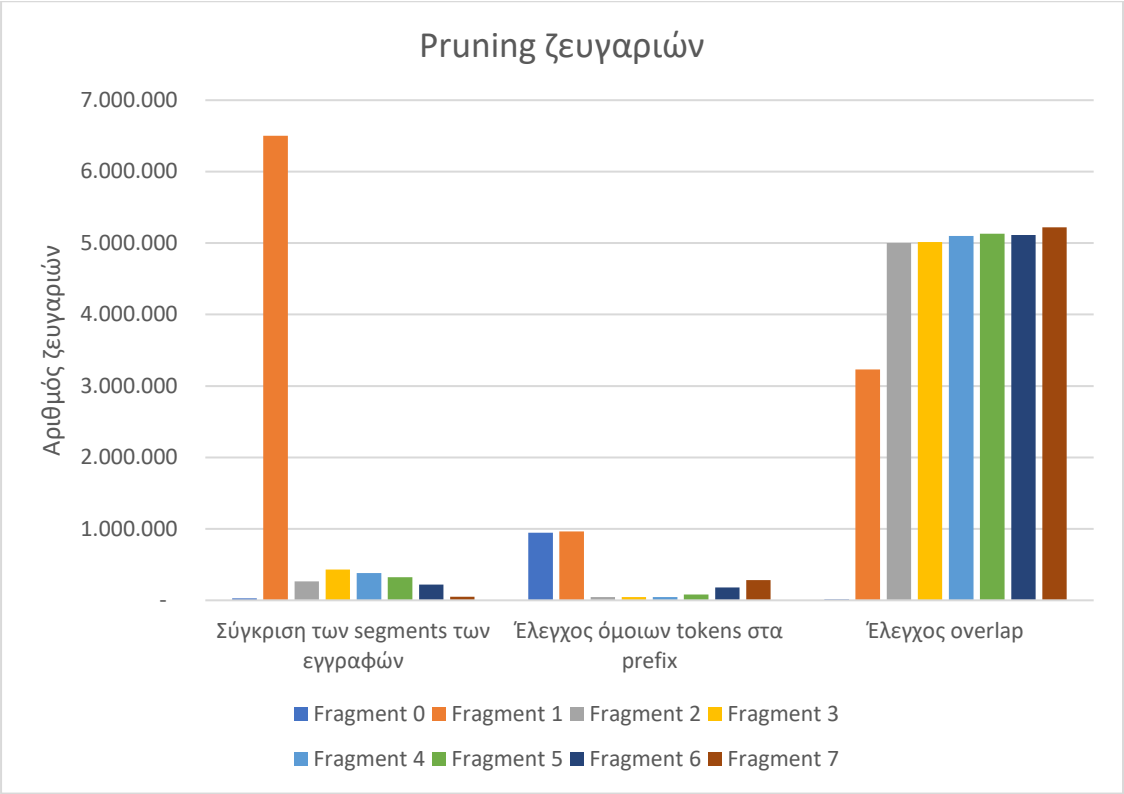
Σχήμα 23. Load Balancing με αλλαγές στον αριθμό των partitions



Σχήμα 24. Ποσοστό του φόρτου με αλλαγές στον αριθμό των partitions

Παρατηρώντας τα Σχήματα 23 και 24, αρχικά γίνεται αντιληπτό ότι όταν τα partitions είναι λίγα, στο πρώτο segment συλλέγονται και μη-σπάνια tokens, με αποτέλεσμα να είναι όλα τα partitions ισορροπημένα. Όμως όσο αυξάνεται ο αριθμός των partitions, υπάρχει το πλεονέκτημα ότι πραγματοποιείται μεγαλύτερος διαμοιρασμός των tokens σε διαφορετικά segments, κάτι το οποίο έχει σαν αποτέλεσμα, ο τελικός φόρτος σε κάθε partition να γίνεται όλο και πιο μικρός. Στην περίπτωση όπου ο αριθμός των partition ήταν 4, ο φόρτος ήταν της τάξης του 82%, ενώ όταν ο αυτός ο αριθμός ήταν 12, ο φόρτος έφτανε σε επίπεδα 75% σε σύγκριση με το αρχικό σύνολο δεδομένων.

Ακόμα ένα βασικό χαρακτηριστικό των text partitioning τεχνικών είναι το, σε πρώιμο στάδιο, ασφαλές φιλτράρισμα (Pruning – κλάδεμα) ζευγαριών τα οποία δεν καλύπτουν τις προϋποθέσεις για το κατώφλι ομοιότητας που έχει τεθεί. Όπως περιεγράφηκε στο Κεφάλαιο 3, αναπτύχθηκαν 3 σταθμοί ελέγχου που περιλαμβάνουν έλεγχο στα segments και στα prefix. Τα αποτελέσματα για κατώφλι ομοιότητας 0,5 και αριθμό partitions 8 εμφανίζονται στο Σχήμα 25.



Σχήμα 25. Pruning ζευγαριών

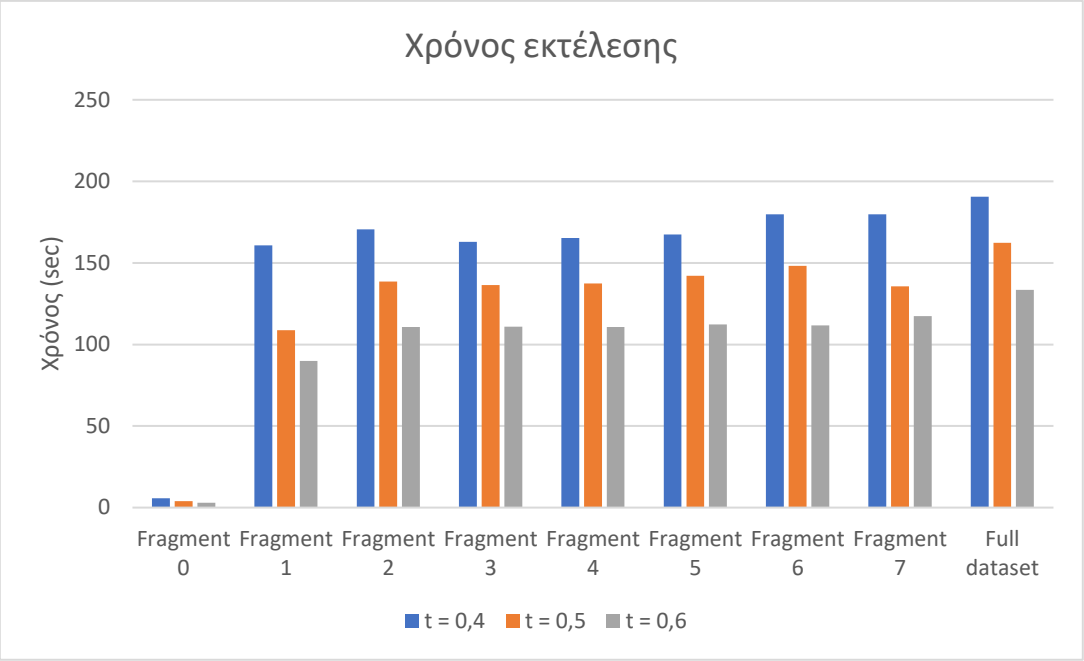
Το Σχήμα 25 υποδεικνύει τα εξής:

Φίλτρο σύγκρισης των segments των εγγραφών: Το πρώτο fragment περιέχει μικρή ποσότητα segments με αποτέλεσμα να μην βρίσκει εφαρμογή αυτό το φίλτρο. Το δεύτερο fragment έχει ικανό αριθμό segments, αλλά αποτελείται από σχετικά σπάνια tokens. Αποτέλεσμα αυτού είναι να κλαδεύονται πολλά ζευγάρια. Για τα υπόλοιπα partitions, εκτός του τελευταίου, υπάρχει μια ομοιόμορφη συμπεριφορά. Στο τελευταίο, υπάρχουν τα πιο «δημοφιλή» tokens με αποτέλεσμα να υπάρχει μεγάλη ομοιότητα, με αποτέλεσμα να μην κλαδεύονται πολλά.

Φίλτρο ελέγχου όμοιων tokens στα prefix: Το prefix περιέχει τα πρώτα tokens των κειμένων, τα οποία tokens είναι σπάνια, και το οποίο prefix είναι μεγαλύτερο από το πρώτο segment. Το αποτέλεσμα είναι να αυξάνεται η πιθανότητα να βρεθούν ανομοιότητες μεταξύ εγγραφών που ανήκουν στα δύο πρώτα fragments.

Έλεγχος overlap: Δεν κλαδεύονται πολλά ζευγάρια του fragment 0 μιας και έχουν μείνει έτσι και αλλιώς ελάχιστα. Αντιθέτως, είναι ένα κρίσιμο σημείο όπου γίνεται pruning για πολλά ζευγάρια των επόμενων partitions.

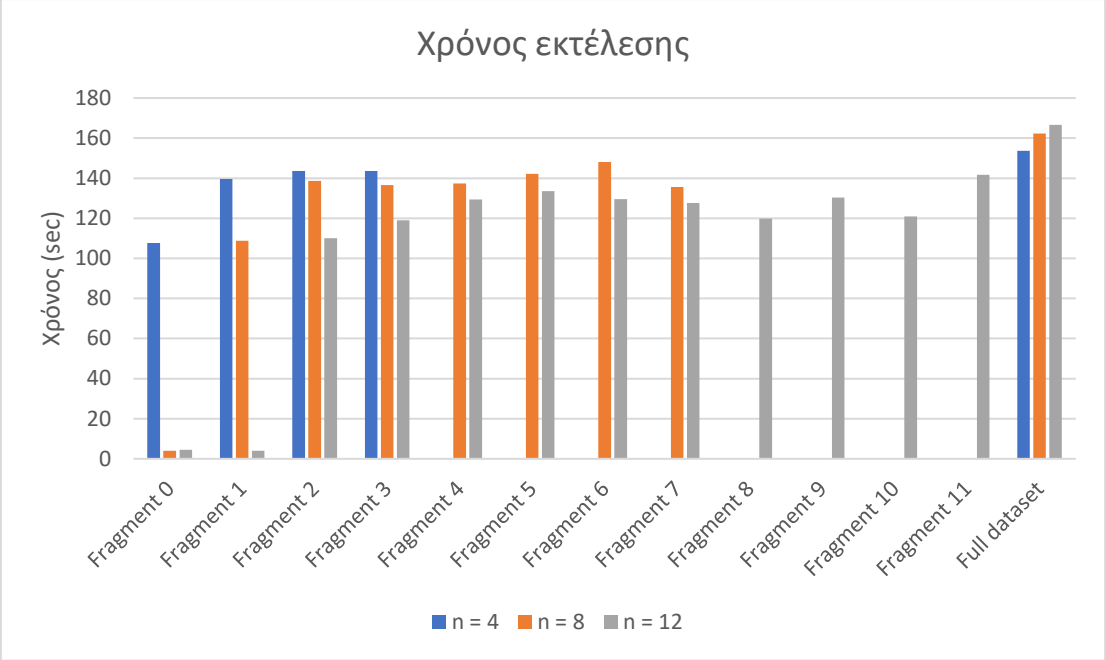
Τέλος, χρονομετρήθηκαν οι χρόνοι εκτέλεσης όπως παρουσιάζεται στο Σχήμα 26.



Σχήμα 26. Χρόνος εκτέλεσης για διαφορετικά κατώφλια ομοιότητας

Με εξαίρεση το πρώτο fragment και για τους λόγους που αναλύθηκαν προηγουμένως οσον αφορά τον μειωμένο αριθμό εγγραφών του, οι χρόνοι για κάθε partition είχαν κοντινές τιμές, και πάντα το μεγαλύτερο partition είχε μικρότερο χρόνο εκτέλεσης σε σχέση με τον χρόνο που θα χρειαζόταν το σύνολο δεδομένων να εκτελεστεί αν δεν είχε υποστεί partitioning. Ακόμα, παρατηρήθηκε ένα μοτίβο που σχετίζεται με την μεταβολή του κατωφλιού ομοιότητας. Όσο αυξάνεται η τιμή του κατωφλιού, τόσο μειώνεται ο χρόνος εκτέλεσης. Η εξήγηση για αυτό είναι ότι όσο πιο απαιτητικό το κατώφλι, τόσο περισσότερα ζευγάρια κλαδεύονται (pruning) και εν τέλει τόσο λιγότερα επεξεργάζονται. Αποτέλεσμα αυτών, είναι χαμηλότεροι χρόνοι επεξεργασίας.

Επίσης έγινε χρονομέτρηση για αλλαγή της τιμής του επιθυμητού αριθμού partitions του συνόλου.



Σχήμα 27. Χρόνοι εκτέλεσης για διαφορετικό αριθμό partitions

Στο Σχήμα 27 παρατηρούμε ότι οι χρόνοι είναι ανάλογοι με το Load balancing για κάθε σενάριο αριθμού partitions, με αυτούς να μειώνονται μιας και μειώνεται και ο αριθμός των εγγραφών για κάθε fragment για τους λόγους που προαναφέρθηκαν. Επίσης έχουν κοντινές τιμές για κάθε fragment, και επίσης πάντα ο μεγαλύτερος χρόνος fragment είναι μικρότερος από αυτόν που θα χρειαζόταν για να τρέξει το πλήρες σύνολο κεντρικοποιημένα. Παρόλο βέβαια που πάντα είναι μικρότερος, μια αρνητική παρατήρηση είναι το γεγονός πως ο χρόνος επεξεργασίας είναι παρόμοιος με την περίπτωση κεντρικοποιημένης επεξεργασίας. Η αιτία για αυτό είναι το ότι η μείωση του φόρτου είναι κοντά στα επίπεδα του 20% μόνο, σε σύγκριση με το πλήρες σύνολο δεδομένων.

4.2.2 Spatial-first με συνθετικό σύνολο δεδομένων

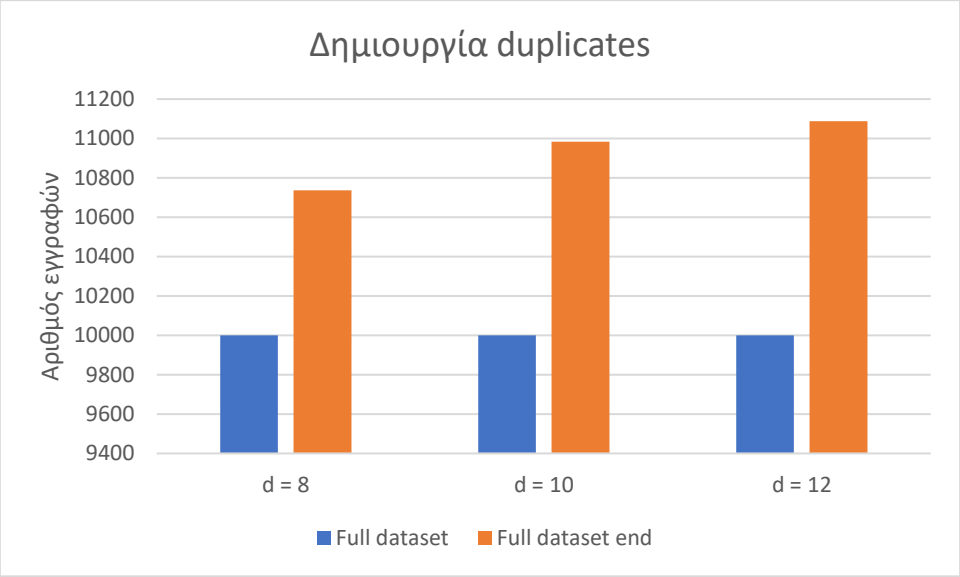
Αυτή η μέθοδος ασχολείται με το το στίγμα των εγγραφών στο χώρο και αφορά τον διαχωρισμό τους βάση αυτού. Οι δοκιμές, έγιναν σε 10.000 εγγραφές και υπήρξαν οι εξής παραλλαγές σε σημαντικές μεταβλητές της διαδικασίας:

d	P	t
8	8	0,5
10	8	0,5
12	8	0,5
10	4	0,5
10	8	0,5
10	12	0,5

Όπου:

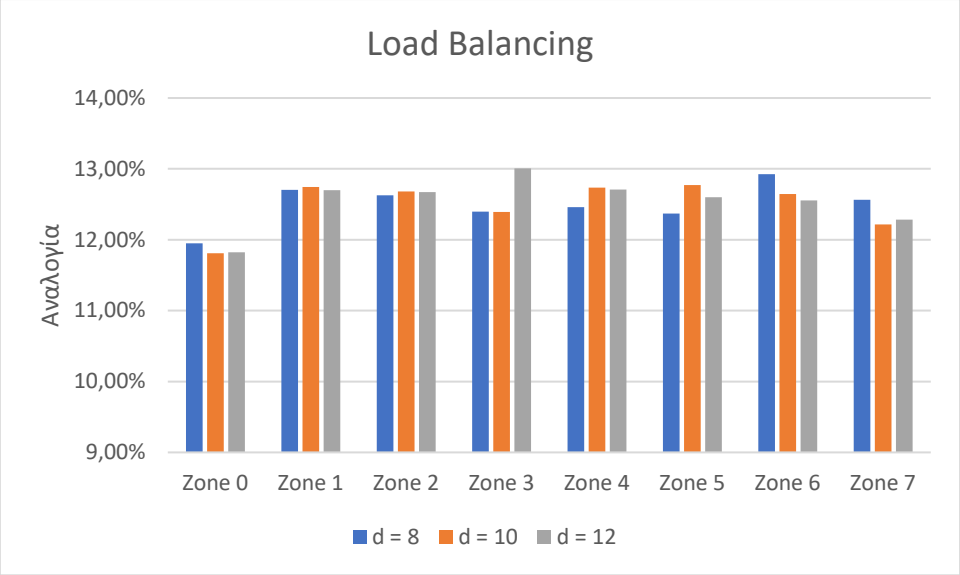
- d: κατώφλι απόστασης
- P: αριθμός Partitions
- t: κατώφλι ομοιότητας. Δεν μεταβλήθηκε στα spatial-first πειράματα μιας και το επίκεντρό αυτών δεν είναι η κειμενική ομοιότητα.

Ομοίως με προηγουμένως, σε πρώτη φάση ερευνήθηκε η κατανομή του φόρτου στα partitions. Αρχικά ελέγχουμε για P = 8, πόσα duplicates δημιουργήθηκαν πάνω στο αρχικό σύνολο δεδομένων, κατά το στάδιο της προεπεξεργασίας.

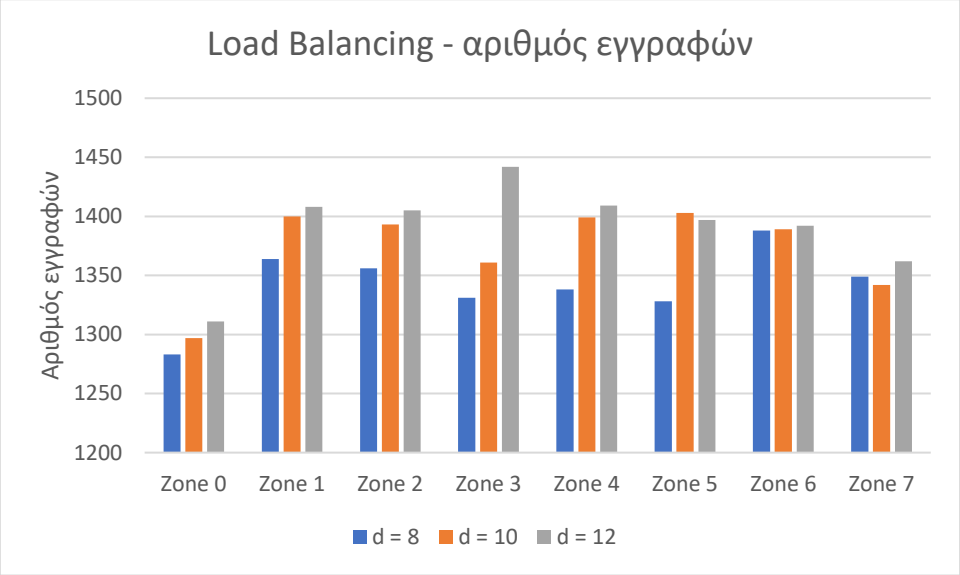


Σχήμα 28. Δημιουργία duplicates για P = 8

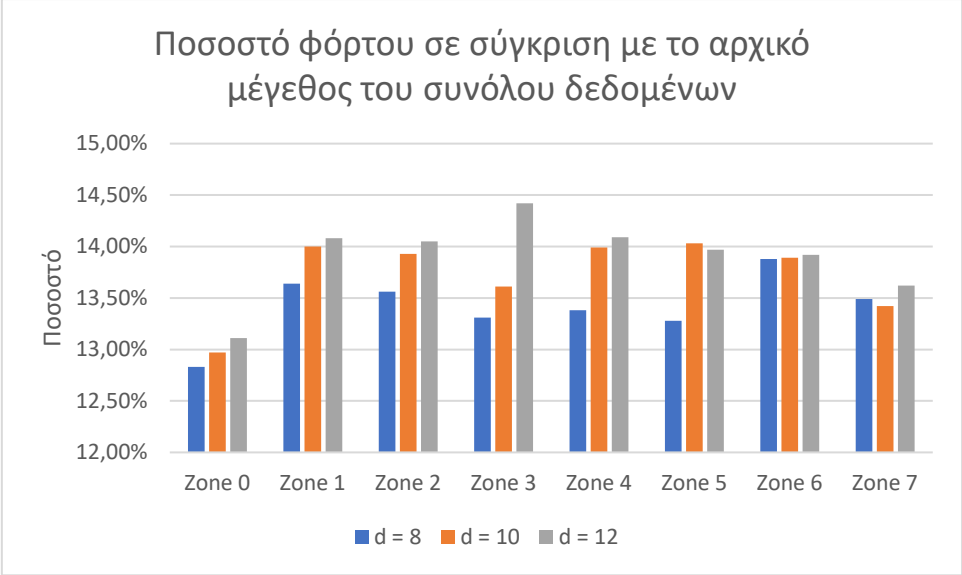
Από το Σχήμα 28 συμπεραίνουμε ότι όσο μεγαλύτερο το κατώφλι απόστασης, τόσο περισσότερα duplicates δημιουργούνται. Αυτό είναι κάτι αναμενόμενο, μιας και η μέθοδος προσπαθεί να εξασφαλίσει ότι, εγγραφές που έχουν στίγματα κοντά στα όρια των ζωνών, πρέπει να μπορέσουν να βρουν το ζεύγος τους αν αυτό βρίσκεται στην άλλη μεριά του συνόρου των ζωνών. Ο αριθμός των duplicates που δημιουργούνται, όπως θα φανεί στη συνέχεια, δεν θεωρείται ανησυχητικός, μιας και η μέθοδος που ερευνάμε μπορεί να εξασφαλίσει πολύ καλή ισορροπία τελικής κατανομής των εγγραφών.



Σχήμα 29. Load Balancing για διάφορα κατώφλια απόστασης



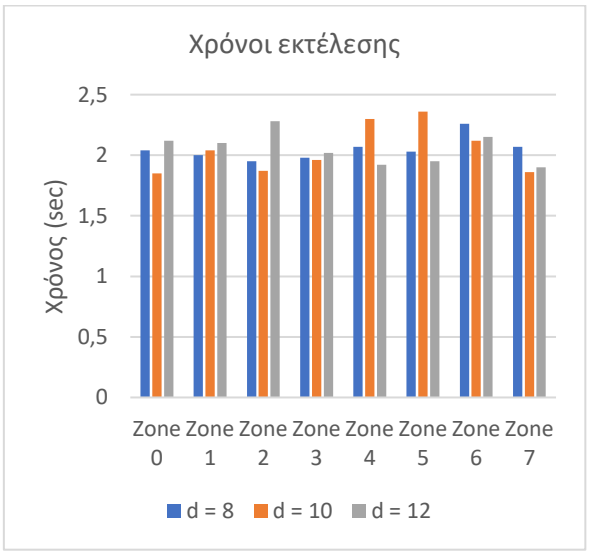
Σχήμα 30. Load Balancing – αριθμός εγγραφών, για διάφορα κατώφλια απόστασης



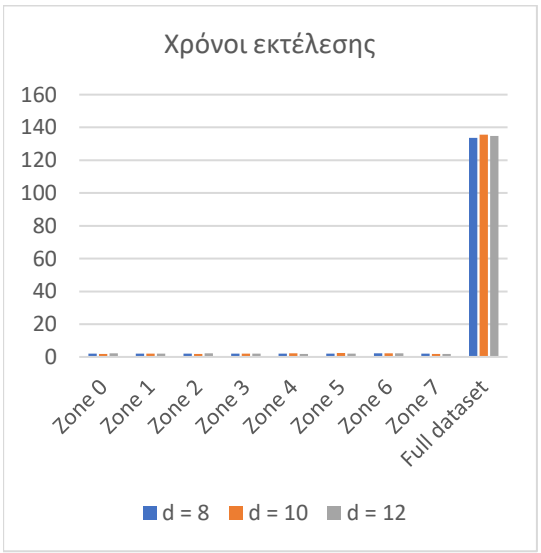
Σχήμα 31. Ποσοστό φόρτου φόρτου για P = 8, για διαφορετικές τιμές του d

Οι μετρήσεις που παρουσιάζονται στα Σχήματα 29, 30 και 31, υποδεικνύουν ότι με τη μέθοδο που χρησιμοποιήσαμε, υπήρξε πολύ καλή εξισορρόπηση για κάθε αλλαγή της τιμής του κατωφλιού απόστασης. Επιπλέον, παρατηρείται ότι για κάθε partition, ο φόρτος του, όταν συγκριθεί με το μέγεθος του αρχικού συνόλου δεδομένων, αγγίζει ένα ποσοστό της τάξης του 14%, κάτι το οποίο εκμεταλλεύεται σε μεγάλο βαθμό τα πλεονεκτήματα των κατανεμημένων συστημάτων επεξεργασίας.

Όσον αφορά τους χρόνους εκτέλεσης:



Σχήμα 32. Χρόνοι εκτέλεσης για διάφορα κατώφλια απόστασης

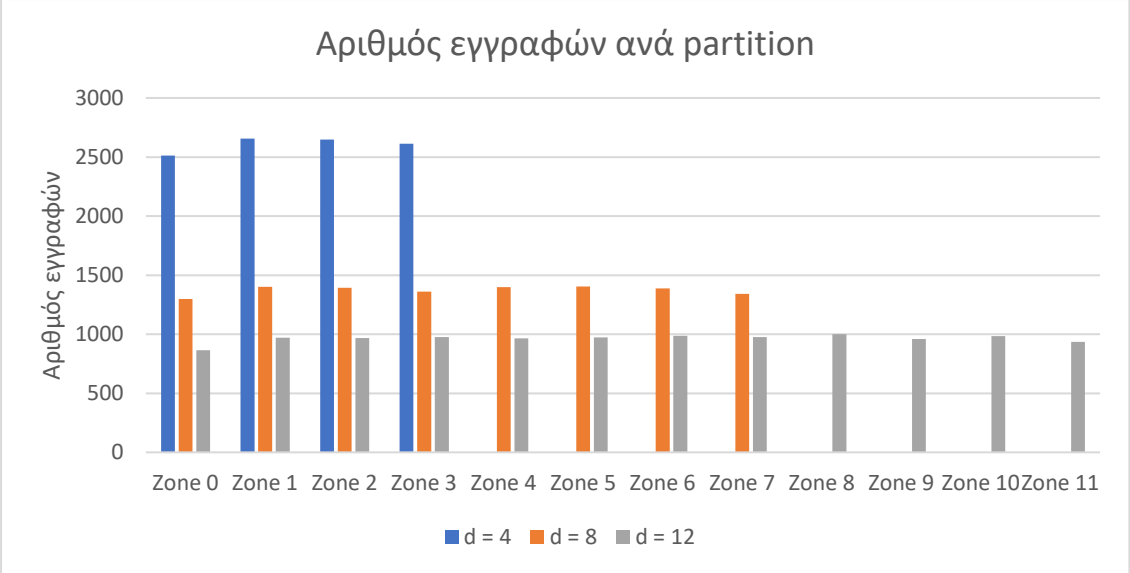


Σχήμα 33. Σύγκριση χρόνων με το πλήρες σύνολο

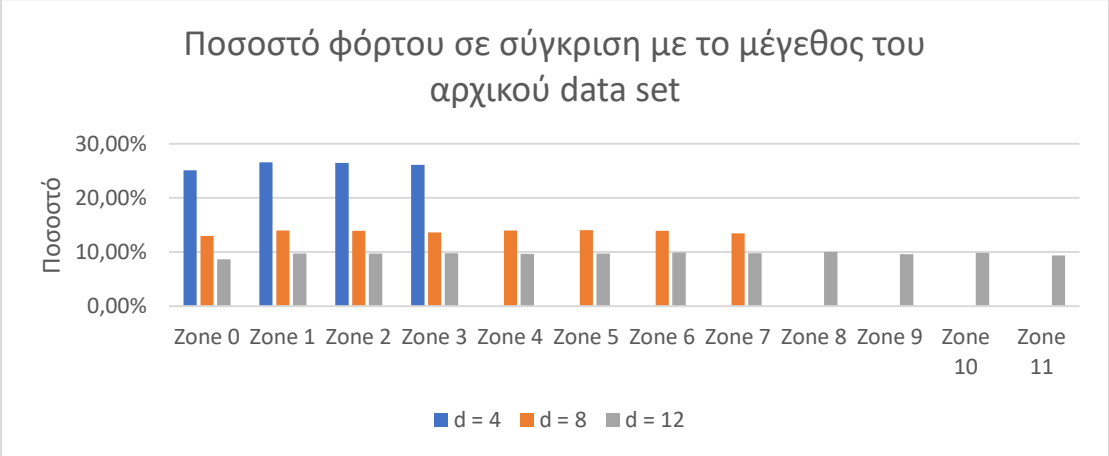
Παρατηρούμε στο Σχήμα 32 ότι είναι ισορροπημένοι αλλά και μικροί, απόρροια της καλής κατανομής και διαχωρισμού των δεδομένων. Από το Σχήμα 33 γίνεται αντιληπτό ότι το

κέρδος είναι μεγάλο, όταν το συγκρίνουμε με το χρόνο που χρειάζεται η επεξεργασία του πλήρους συνόλου δεδομένων.

Δοκιμάζοντας παραλλαγές στην τιμή του αριθμού των partitions, παραλαμβάνουμε τα παρακάτω αποτελέσματα:

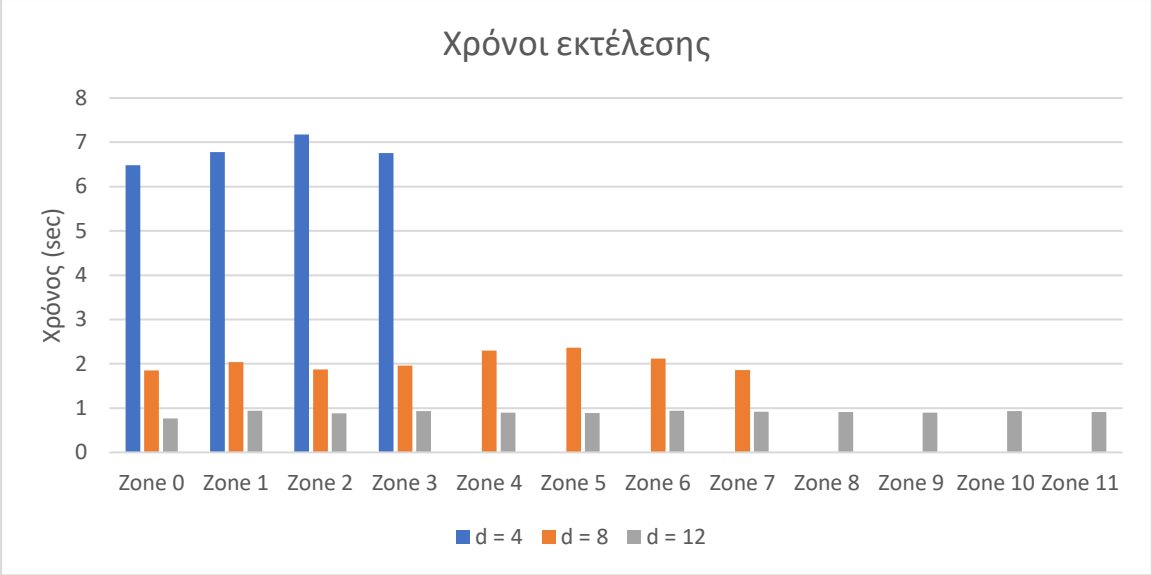


Σχήμα 34. Αριθμός εγγραφών ανά partition για d ποσότητα partitions



Σχήμα 35. Ποσοστό φόρτου ανά partition για d διαφορετική ποσότητα partitions

Όπως βλέπουμε στα Σχήματα 34 και 35, η κατανομή εξακολουθεί να είναι πολύ καλή. Αρχικά, σταθερά για κάθε μεταβολή του αριθμού των partitions, υπάρχει ισορροπία, και επίσης, όσο περισσότερα είναι τα partitions, τόσο μεγαλύτερο είναι το κέρδος όσον αφορά τον φόρτο. Το οφέλη είναι ξεκάθαρα και η spatial-first υλοποίηση δείχνει να πλεονεκτεί έναντι της text-first.



Σχήμα 36. Χρόνοι εκτέλεσης ανά partition για d διαφορετική ποσότητα partitions

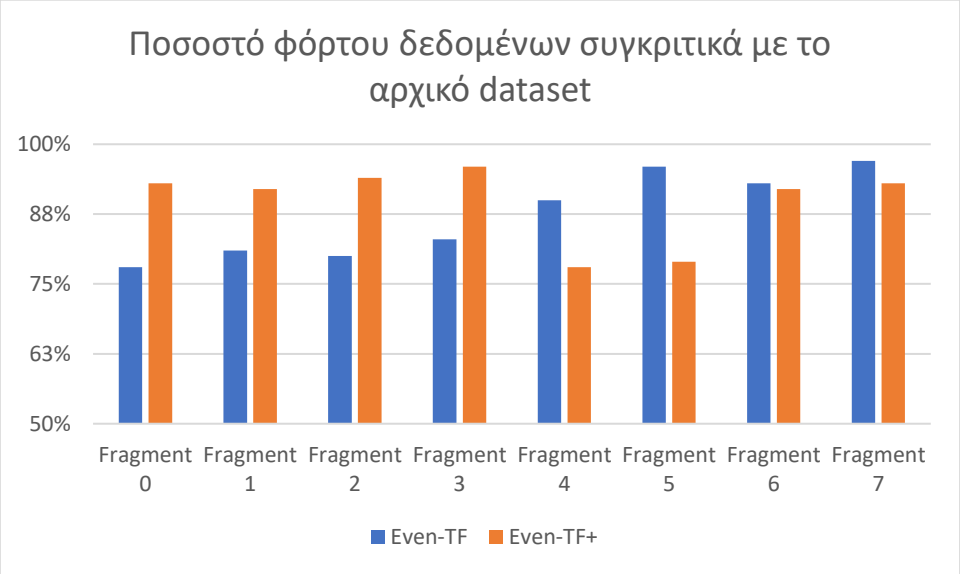
Αντιστοίχως, ο χρόνος μειώνεται όσο αυξάνονται τα partitions, μιας και έτσι η διαδικασία επεξεργάζεται λιγότερες εγγραφές (Σχήμα 36).

4.2.3 Text-first με πραγματικό σύνολο δεδομένων

Υπήρξαν οι εξής παραλλαγές σε σημαντικές μεταβλητές της διαδικασίας:

t	P	d
0,7	8	10
0,8	8	10
0,9	8	10
0,8	4	10
0,8	8	10
0,8	12	10

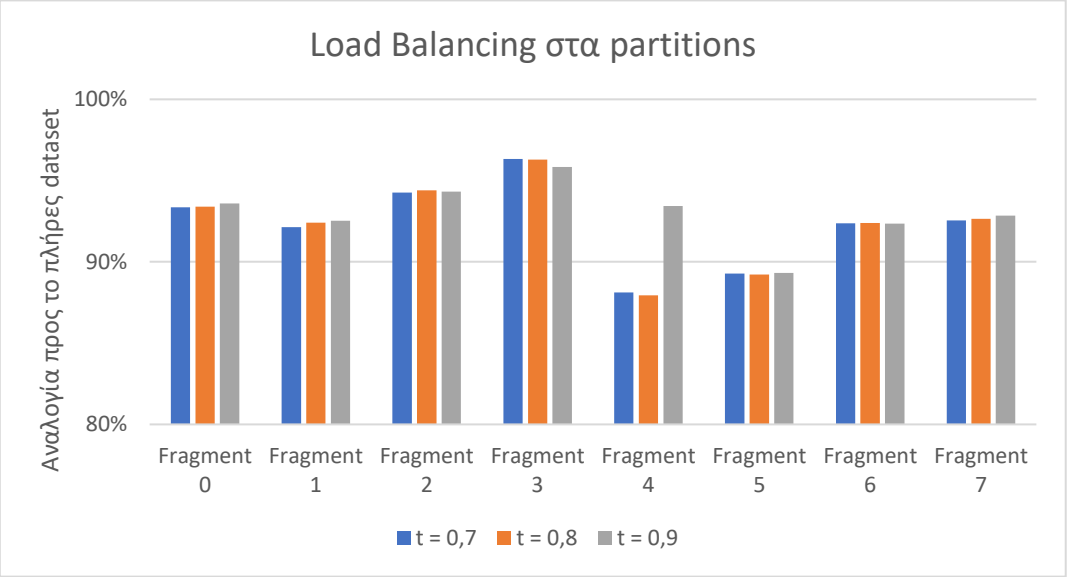
Όπως περιεγράφηκε στο Κεφάλαιο 3, για την εξισορρόπηση του φόρτου έγινε χρήση της μεθόδου Even-TF για την επιλογή των pivots που θα καθορίσουν τα όρια των segments [8]. Παρ' όλα αυτά, και αναλόγως τη σύσταση ενός συνόλου δεδομένων, υπάρχει η πιθανότητα να υπάρχουν μεγάλες αποκλίσεις στην κατανομή των tokens αν ακολουθηθεί αυτή η μέθοδος με αυστηρό τρόπο, όπως βλέπουμε στο Σχήμα 37 (8 partitions, t = 0,8).



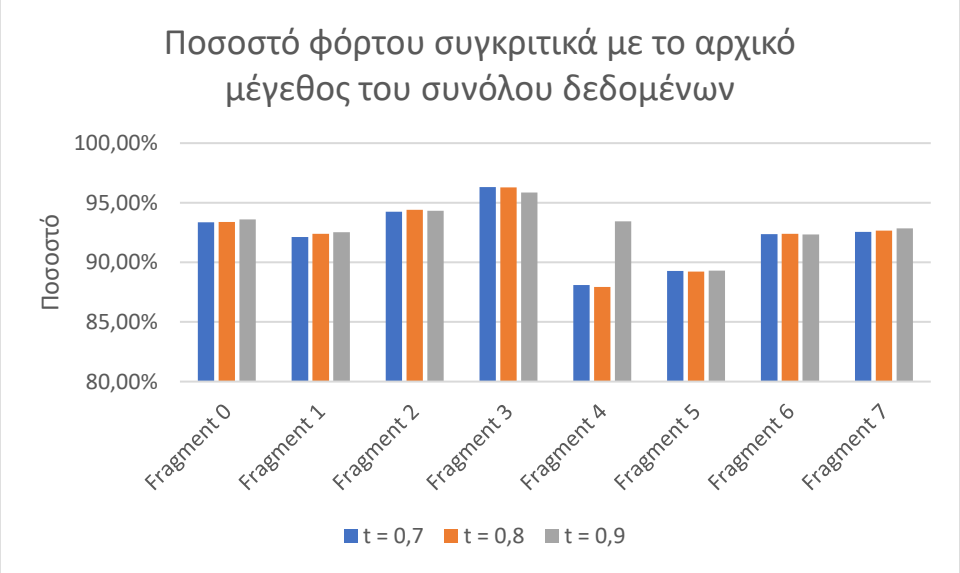
Σχήμα 37. Ποσοστό φόρτου

Λόγω των παραπάνω, κατά τη διαδικασία των πειραμάτων για το συγκεκριμένο σύνολο δεδομένων, ακολουθήθηκε μια λογική (που για τις ανάγκες της παρούσας εργασίας της δίνεται μια ενδεικτική ονομασία Event-TF+) η οποία πατάει στην Even-TF, αλλά έγιναν κάποιες μεταβολές με την έννοια ότι μεταβλήθηκαν αυξητικά τα όρια των ποσοστημορίων. Θυμίζουμε πως κατά την Even-TF, για παράδειγμα, αν θέλουμε 4 partitions και η αθροισμένη συχνότητα μέχρι-ένα-token-πριν το *«A»* token αγγίζει το 50% του αριθμού των tokens του συνόλου δεδομένων, τότε κρατάμε το *«A»* σαν το pivot όπου σε μέγιστο βαθμό σε αυτό θα τελειώνει το 2ο segment σε κάθε εγγραφή, με βάση την ταξινόμηση που

έχουμε δημιουργήσει στα tokens. Η αλλαγή που κάναμε τώρα ήταν να τεθούν τα όρια αυτών των ποσοστημορίων σε μία μεγαλύτερη τιμή προκειμένου να υπάρξει καλύτερη εξισορρόπηση. Για παράδειγμα, για την παραπάνω περίπτωση, το ποσοστό που χρησιμοποιήθηκε δεν ήταν 50% αλλά 60%. Με αυτόν τον τρόπο η εξισορρόπηση δείχνει να βελτιώνεται, αν και αυτό συμβαίνει σε μικρό βαθμό. Το συγκεκριμένο στοιχείο είναι ένα από τα συμπεράσματα της παρούσας Διπλωματικής εργασίας το οποίο αξίζει περαιτέρω διερεύνησης όσον αφορά δηλαδή τον συσχετισμό της σύστασης και ιδιομορφίας ενός συνόλου δεδομένων με την τεχνική Even-TF.

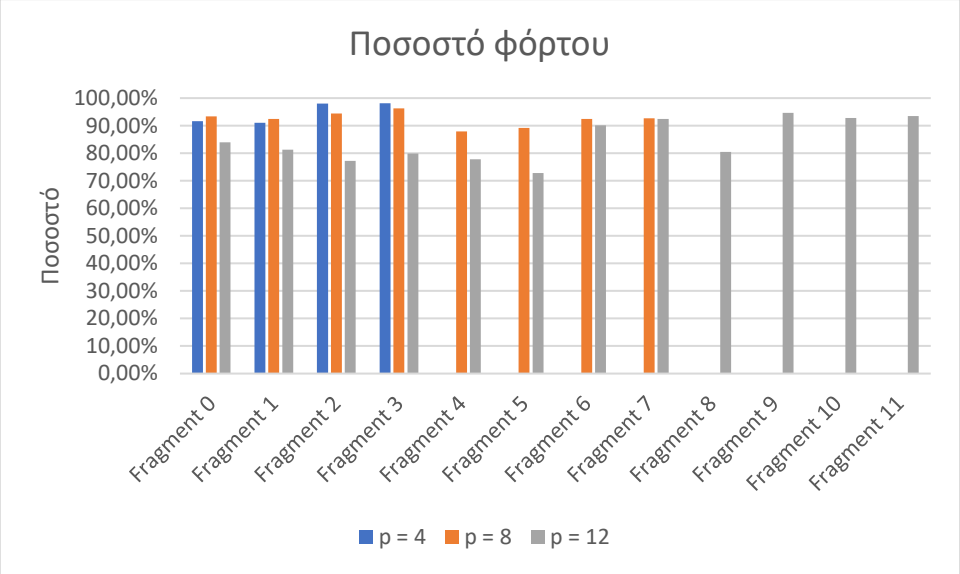


Σχήμα 38. Load Balancing για διαφορετικά κατώφλια ομοιότητας



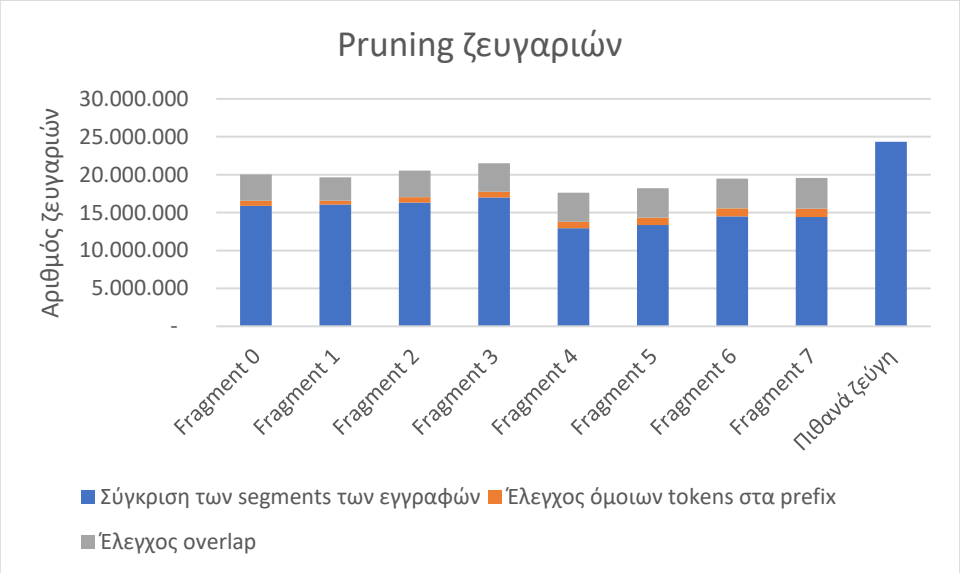
Σχήμα 39. Ποσοστό φόρτου για διαφορετικά κατώφλια ομοιότητας

Όπως παρατηρούμε στα Σχήματα 38 και 39, στην προσομοίωση που διεξήχθη, παρά τη σχετικά καλή εξισορρόπηση, η μείωση του φόρτου των δεδομένων συγκριτικά με το αρχικό μέγεθος του συνόλου δεδομένων ήταν σχετικά μικρή. Παρ' όλα αυτά ήταν υπαρκτή και με αυξητικές τάσεις όσο αυξανόταν ο αριθμός των partitions, όπως παρατηρούμε στο Σχήμα 40.



Σχήμα 40. Ποσοστό φόρτου για διαφορετικό αριθμό partitions

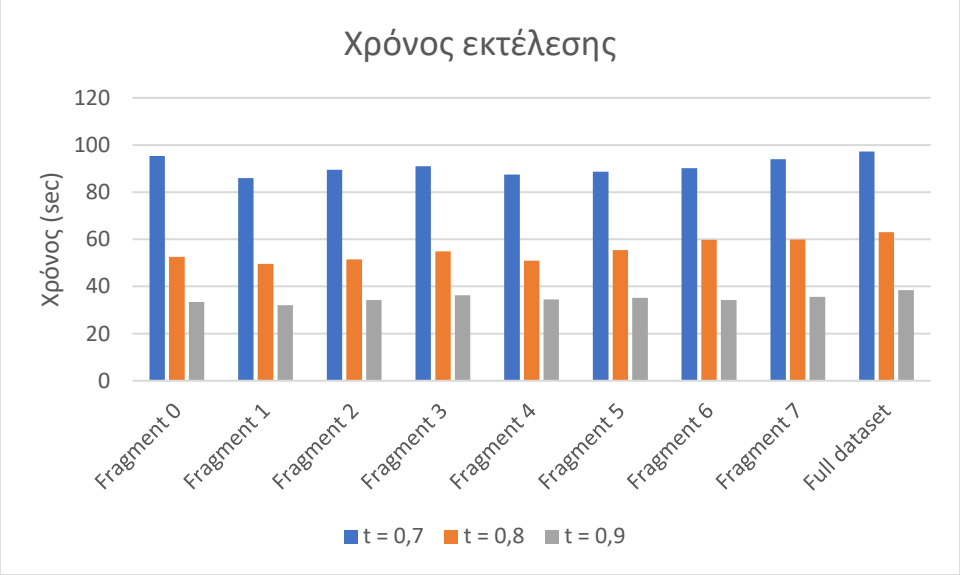
Όσον αφορά το κομμάτι του pruning ζευγαριών, για t = 0,8 , λειτούργησε αποδοτικά απορρίπτοντας την πλειοψηφία των πιθανών ζευγαριών και αφήνοντας τη μειοψηφία σαν υποψήφια ζευγάρια (Σχήμα 41).



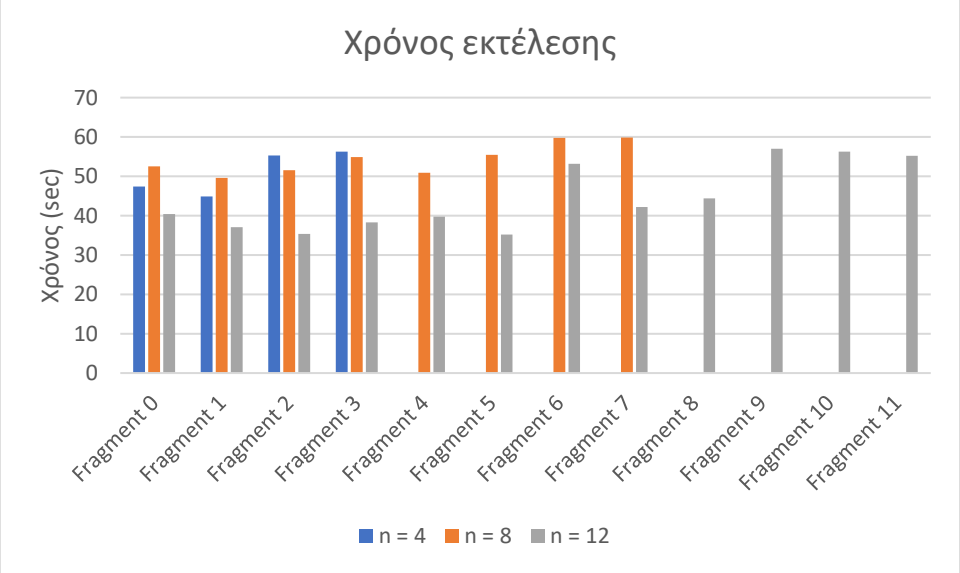
Σχήμα 41. Pruning ζευγαριών

Η παραπάνω συμπεριφορά όσον αφορά τον έλεγχο στον οποίο έγινε ο μεγαλύτερος αριθμός κλαδεμάτων, είναι ομοιόμορφη και διαφέρει από τη συμπεριφορά στο τεχνητό σύνολο δεδομένων μιας και σε αυτή την περίπτωση έχει γίνει παραλλαγή του Even-TF έτσι ώστε να μειωθούν οι μεγάλες αποκλίσεις.

Επιπροσθέτως, όπως παρουσιάζεται στα Σχήματα 42 και 43, όσο αυξάνεται η τιμή του κατωφλιού, τόσο μειώνεται ο χρόνος εκτέλεσης. Η εξήγηση για αυτό είναι ότι όσο πιο απαιτητικό το κατώφλι, τόσο περισσότερα ζευγάρια κλαδεύονται (pruning) και εν τέλει τόσο λιγότερα επεξεργάζονται. Αποτέλεσμα αυτών, είναι χαμηλότεροι χρόνοι επεξεργασίας. Επίσης, όταν αυξάνεται ο αριθμός των Partition, μειώνεται ο χρόνος εκτέλεσης μιας και τα δεδομένα σε κάθε Partition είναι μειωμένα.



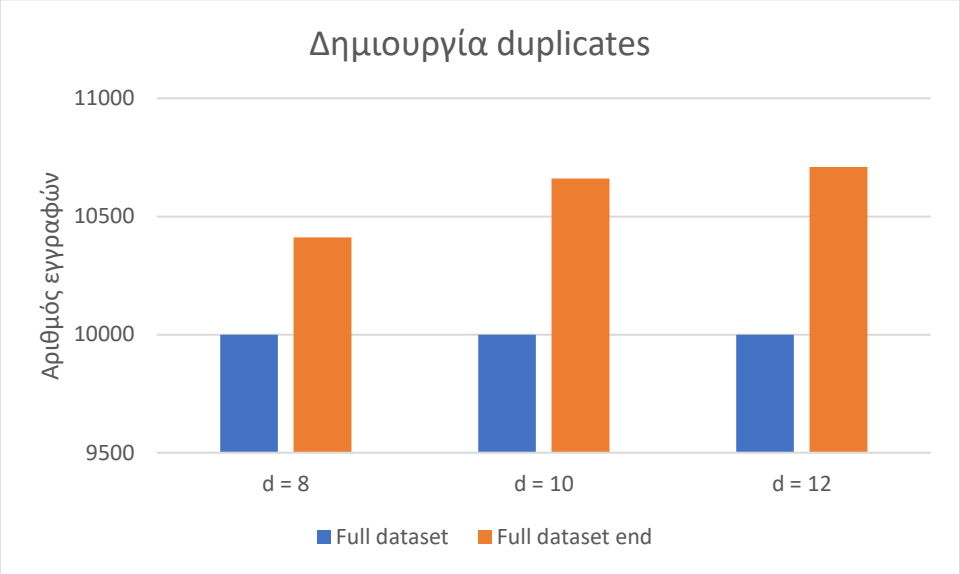
Σχήμα 42. Χρόνοι εκτέλεσης για διαφορετικά κατώφλια απόστασης



Σχήμα 43. Χρόνοι εκτέλεσης για διαφορετικό αριθμό partitions

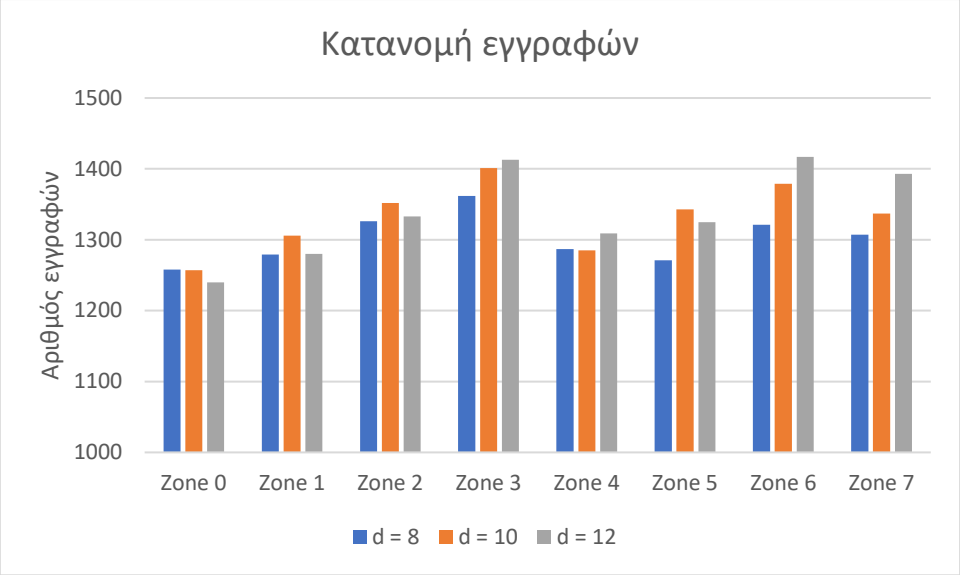
4.2.2 Spatial-first με πραγματικό σύνολο δεδομένων

Η μέθοδος, όπως έχει αναφερθεί συνδυάζεται με τη δημιουργία duplicates κάποιων εγγραφών μέσα στο ίδιο partition. Στο Σχήμα 44 παρατηρούμε ξανά ότι όσο μεγαλύτερο το κατώφλι απόστασης, τόσο περισσότερα duplicates δημιουργούνται. Αυτό είναι κάτι αναμενόμενο, μιας και η μέθοδος προσπαθεί να εξασφαλίσει ότι, εγγραφές που έχουν στίγματα κοντά στα όρια των ζωνών, πρέπει να μπορέσουν να βρουν το ζεύγος τους αν αυτό βρίσκεται στην άλλη μεριά του συνόρου των ζωνών.

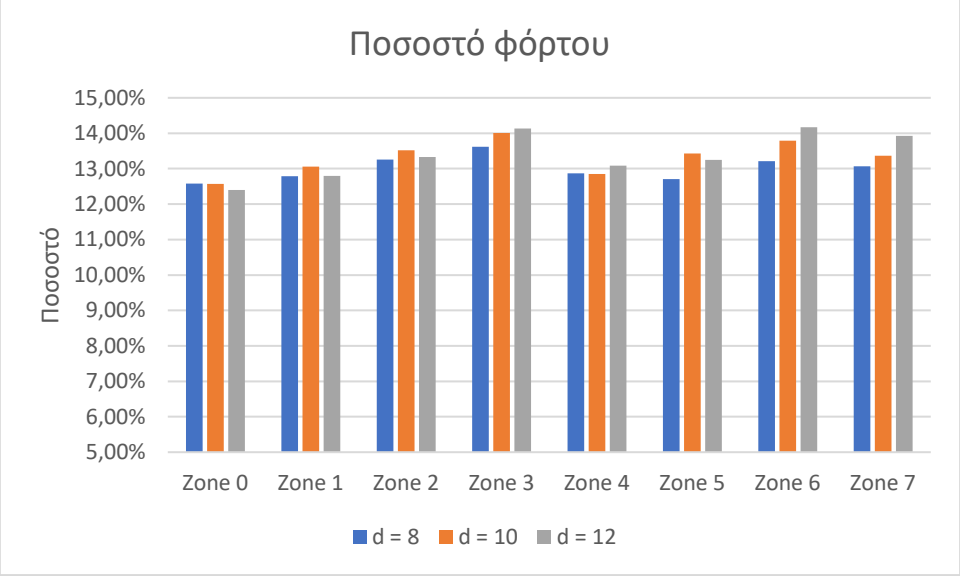


Σχήμα 44. Δημιουργία duplicates

Η κατανομή των εγγραφών είναι εξισορροπημένη (Σχήμα 45) και επίσης υπάρχει μεγάλη μείωση των δεδομένων σε σύγκριση με το αρχικό σύνολο δεδομένων. Άρα το κέρδος μια κατανεμημένης επεξεργασίας είναι μεγάλο. Στο Σχήμα 46 παρατηρούμε τον φόρτο που όντως αυξάνεται όσο μεγαλώνει η τιμή του κατωφλιού απόστασης, μιας και δημιουργούνται περισσότερα διπλότυπα.

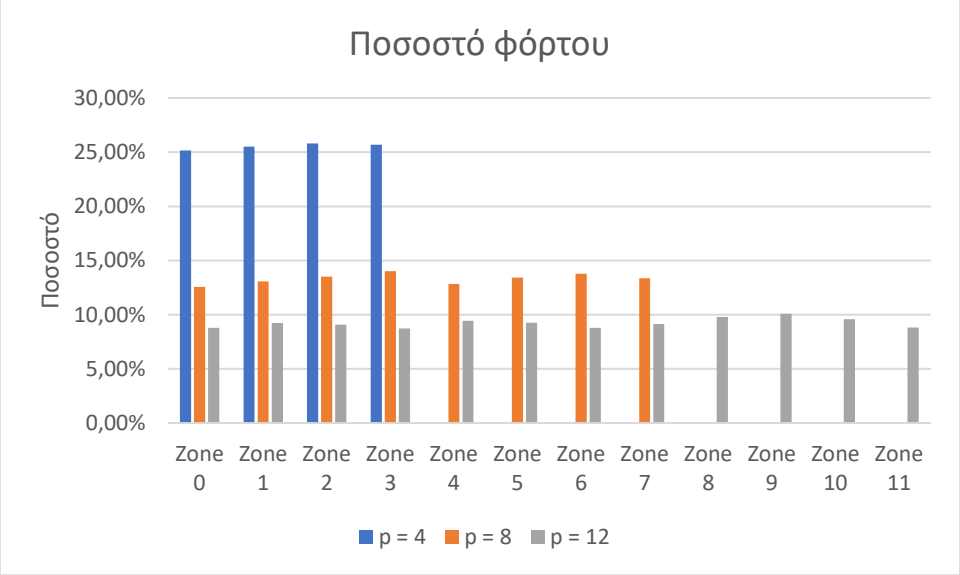


Σχήμα 45. Κατανομή εγγραφών για διαφορετικό κατώφλι απόστασης

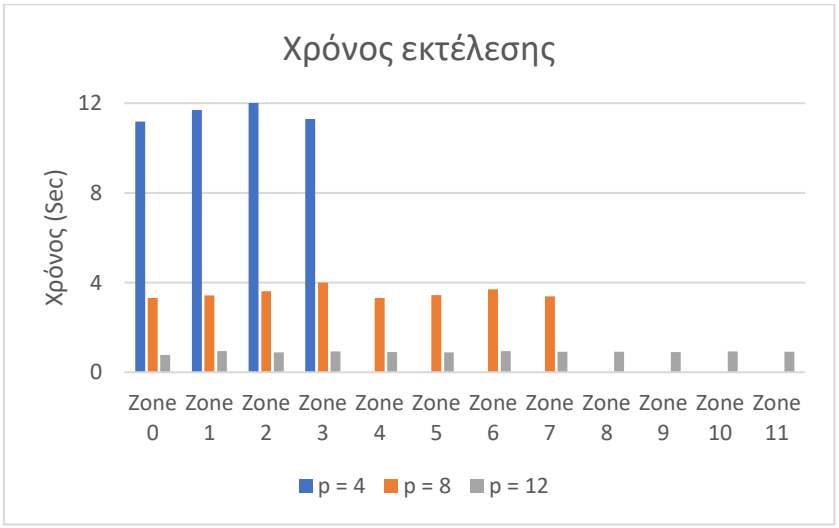


Σχήμα 46. Ποσοστό φόρτου για διαφορετικό κατώφλι απόστασης

Επιπλέον, όσο αυξάνεται ο αριθμός των partitions, τόσο περισσότερο μειώνεται ο φόρτος τους και οπότε ευνοείται η «ελάφρυνση» της επεξεργασίας, όπως βλέπουμε και στο Σχήμα 47. Απόρροια των ανωτέρω είναι το αποτέλεσμα του Σχήματος 48 όπου παρατηρούμε τη μείωση του χρόνου εκτέλεσης όσο αυξάνεται ο αριθμός των partitions. Αξίζει να σημειωθεί ότι ο χρόνος εκτέλεσης για το πλήρες σύνολο δεδομένων άγγιζε τιμές άνω των 200 δευτερολέπτων.



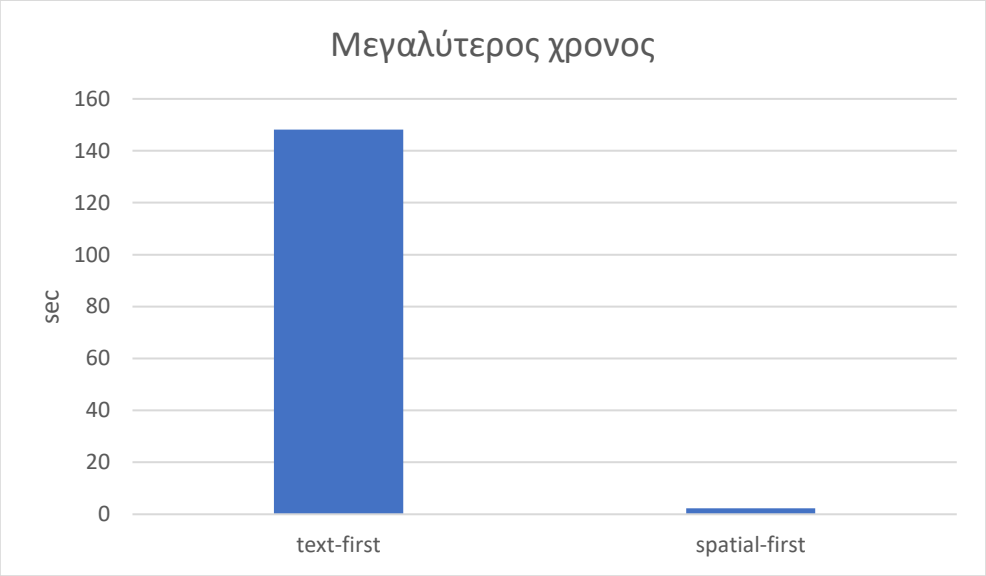
Σχήμα 47. Ποσοστό φόρτου για διαφορετικό αριθμό partitions



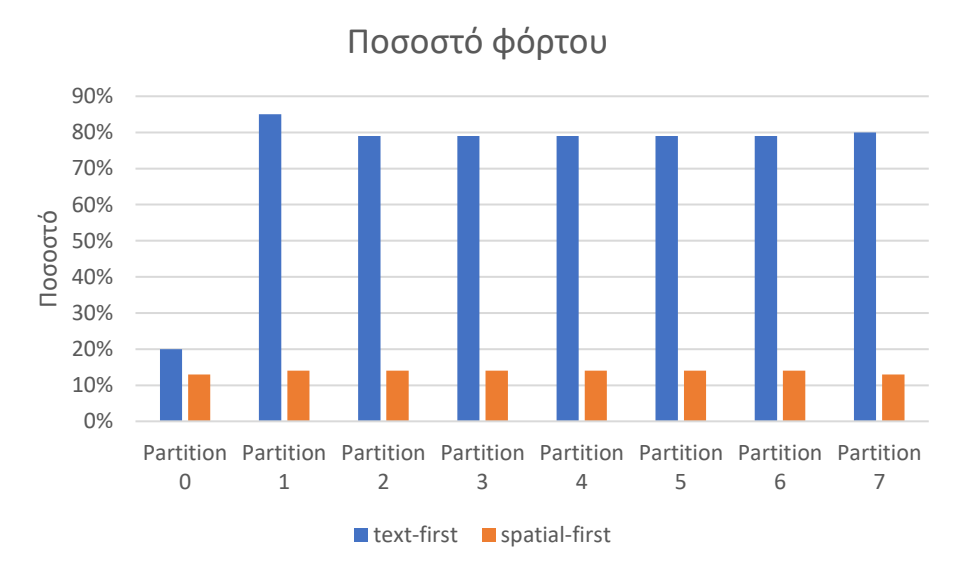
Σχήμα 48. Χρόνοι εκτέλεσης για διαφορετικό αριθμό partitions

4.3 Σύγκριση των τεχνικών

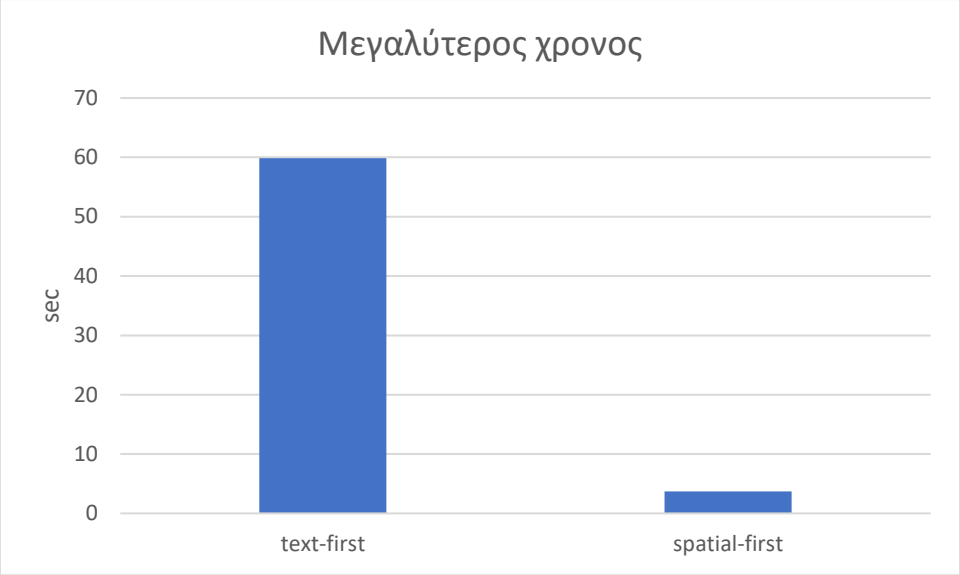
Συγκρίνοντας τα αποτελέσματα των δύο υλοποιήσεων για t = 0,5, d =10 και αριθμό partitions = 8, είναι προφανές ότι για τις συγκεκριμένες τιμές των μεταβλητών, όπως και τις κοντινές αυτών όπως παρουσιάστηκε στα παραπάνω υποκεφάλαια, η spatial-first τεχνική φαίνεται να υπερτερεί της text-first όσον αφορά την απόδοση. Όπως παρατηρούμε (Σχήματα 49, 50, 51 και 52), στην περίπτωση της spatial-first, ο φόρτος στα partitions είναι μικρότερος, όπως και ο χρόνος εκτέλεσης, και για τα δύο είδη των συνόλων δεδομένων (συνθετικό και πραγματικό).



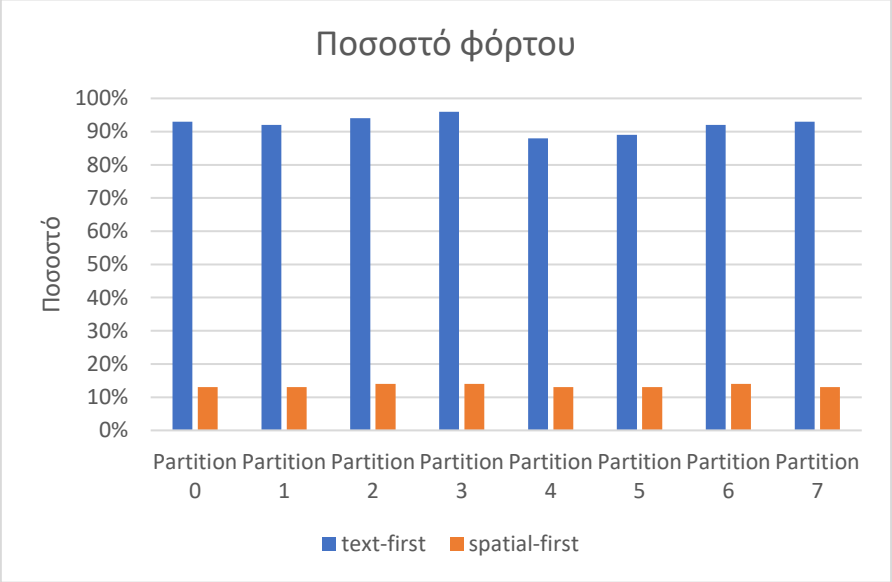
Σχήμα 49. Χρόνος εκτέλεσης – συνθετικό σύνολο δεδομένων



Σχήμα 50. Φόρτος – συνθετικό σύνολο δεδομένων



Σχήμα 51. Χρόνος εκτέλεσης – πραγματικό σύνολο δεδομένων



Σχήμα 52. Φόρτος - πραγματικό σύνολο δεδομένων

5. Συμπεράσματα και μελλοντικές προοπτικές

Στην παρούσα διπλωματική εργασία ερευνήθηκε η διαχείριση χωρο-κειμενικών δεδομένων με επίκεντρο τη σύζευξη στοιχείων βάσει της εγγύτητας της απόστασής τους και της ομοιότητας του κειμενικού τους μέρους, Επίσης ερευνήθηκε η πρότερη κατανομή των δεδομένων προκειμένου να ευνοηθούν διαδικασίες που εκτελούνται από συστήματα κατανεμημένης και παράλληλης επεξεργασίας ερωτημάτων. Αρχικά παρουσιάστηκε μια text-first προσέγγιση, βασισμένη στις δημοσιεύσεις των αλγορίθμων FS-Join [8] και PP-Join [3], η οποία κατανέμει τα δεδομένα με βάση το κειμενικό τους μέρος και το χωρίζει σε κομμάτια (segments) βασισμένη σε pivot tokens που προκύπτουν από την τεχνική Even-TF η οποία διαχειρίζεται ισορροπημένα τη γενική συχνότητα των tokens. Από την πειραματική διαδικασία εξήχθη πως είναι δυνατόν να υπάρξει ισορροπημένη κατανομή στα διάφορα partitions και να μειωθεί ο φόρτος του συνόλου όταν αυτός συγκριθεί με το αρχικό μέγεθος του συνόλου δεδομένων. Η αύξηση του αριθμού των partitions μειώνει ακόμα περισσότερο τον φόρτο. Οι έλεγχοι για pruning (κλάδεμα) πιθανών ζευγαριών συνεισφέρουν στη μείωση του αριθμού υποψηφίων ζευγαριών που τελικά θα επεξεργαστούν. Αναλόγως τη μορφολογία των δεδομένων, υπάρχει η πιθανότητα να χρειάζεται παραλλαγή των παραμέτρων της τεχνικής Even-TF, έτσι ώστε να βελτιωθεί η ισορροπία του φόρτου. Έπειτα παρουσιάστηκε μια spatial-first προσέγγιση η οποία κατανέμει τα δεδομένα με βάση το στίγμα της θέσης τους. Αυτή η προσέγγιση φτάνει σε μία κατανομή πολύ καλού επιπέδου εξισορρόπησης και μεγάλης μείωσης του φόρτου και των χρόνων εκτέλεσης. Επίσης παρατηρήθηκε ότι όσο αυξάνεται ο αριθμός των partitions και όσο αυξάνεται το κατώφλι απόστασης, δημιουργούνται περισσότερα duplicates, κάτι το οποίο μας οδηγεί στο συμπέρασμα πως παρά την γενικά εξαιρετική απόδοση της τεχνικής, μετά από κάποια τιμή των παραπάνω μεταβλητών μπορεί να μην είναι αποδοτική. Τέλος μπορούμε να σημειώσουμε πως κατά την text-first προσέγγιση όσο αυξάνεται ο αριθμός των partitions, μειώνεται ο φόρτος τους.

Μελλοντικά θα μπορούσαν να υλοποιηθούν οι τεχνικές σε κατανεμημένα συστήματα επεξεργασίας μεγάλων δεδομένων όπως το Apache Spark, καθώς και να ελεγχθεί η συμπεριφορά των τεχνικών για ακραίες τιμές του αριθμού των partitions και των κατωφλιών απόστασης και ομοιότητας μιας και παρατηρήθηκε ότι αυτές οι μεταβολές επηρεάζουν τον φόρτο που κατανέμεται σε κάθε partition. Αξίζει ειδικά να μελετηθεί η συμπεριφορά της text-first προσέγγισης - η οποία στην προσομοίωση της παρούσας εργασίας φαίνεται ότι μειονεκτεί - για πολύ μεγάλο αριθμό partitions μιας και οι ενδείξεις που προκύπτουν από την πειραματική διαδικασία δείχνουν ότι ο φόρτος των partitions έχει πτωτικές τάσεις με την αύξηση των partition. Αντιστοίχως αξίζει να μελετηθεί η συμπεριφορά της spatial-first - η οποία στην προσομοίωση της παρούσας εργασίας φαίνεται ότι πλεονεκτεί - για μεγάλες τιμές του κατωφλιού απόστασης, μιας και η πειραματική διαδικασία έδειξε ότι σε αυτές τις περιπτώσεις παράγονται πολλά διπλότυπα σε κάθε partition και επομένως ο φόρτος θα μεγαλώσει. Τέλος, θα μπορούσε να ερευνηθεί μια βέλτιστη διαμόρφωση της τεχνικής Even-TF σε σχέση με την ιδιομορφία και τη σύσταση ενός συνόλου δεδομένων.

6. Βιβλιογραφία

[1] Jinfeng Rao, Jimmy J. Lin, Hanan Samet: Partitioning strategies for spatio-textual similarity join. BigSpatial@SIGSPATIAL 2014: 40-49

[2] Roberto J. Bayardo, Yiming Ma, Ramakrishnan Srikant: Scaling up all pairs similarity search. WWW 2007: 131-140

[3] Chuan Xiao, Wei Wang, Xuemin Lin, Jeffrey Xu Yu: Efficient similarity joins for near duplicate detection. WWW 2008: 131-140

[4] Willi Mann, Nikolaus Augsten, Panagiotis Bouros: An Empirical Evaluation of Set Similarity Join Techniques. Proc. VLDB Endow. 9(9): 636-647 (2016)

[5] Panagiotis Bouros, Shen Ge, Nikos Mamoulis: Spatio-textual similarity joins. Proc. VLDB Endow. 6(1): 1-12 (2012)

[6] Jiannan Wang, Guoliang Li, Jianhua Feng: Can we beat the prefix filtering?: an adaptive framework for similarity join and search. SIGMOD Conference 2012: 85-96

[7] Fabian Fier, Nikolaus Augsten, Panagiotis Bouros, Ulf Leser, Johann-Christoph Freytag: Set Similarity Joins on MapReduce: An Experimental Survey. Proc. VLDB Endow. 11(10): 1110-1122 (2018)

[8] Chuitian Rong, Chunbin Lin, Yasin N. Silva, Jianguo Wang, Wei Lu, Xiaoyong Du: Fast and Scalable Distributed Set Similarity Joins for Big Data Analytics. ICDE 2017: 1059-1070

[9] Ziad Sehili, Lars Kolb, Christian Borgs, Rainer Schnell, Erhard Rahm: Privacy Preserving Record Linkage with PPJoin. BTW 2015: 85-104

[10] Dong Deng, Guoliang Li, Shuang Hao, Jiannan Wang, Jianhua Feng: MassJoin: A mapreduce-based method for scalable string similarity joins. ICDE 2014: 340-351

[11] Rares Vernica, Michael J. Carey, Chen Li: Efficient parallel set-similarity joins using MapReduce. SIGMOD Conference 2010: 495-506

[12] Dong Deng, Guoliang Li, He Wen, Jianhua Feng: An Efficient Partition Based Method for Exact Set Similarity Joins. Proc. VLDB Endow. 9(4): 360-371 (2015)

[13] Shuyao Qi, Panagiotis Bouros, Nikos Mamoulis: Efficient Top-k Spatial Distance Joins. SSTD 2013: 1-18

[14] Raphael A. Finkel, Jon Louis Bentley: Quad Trees: A Data Structure for Retrieval on Composite Keys. Acta Informatica 4: 1-9 (1974)