ΔΙΑΧΕΙΡΙΣΗ ΣΥΝΘΕΤΩΝ ΔΕΔΟΜΕΝΩΝ
4Η ΕΡΓΑΣΙΑ

ΣΤΟΧΟΣ ΕΡΓΑΣΙΑΣ

Σε αυτή την εργασία καλούμαστε να υλοποιήσουμε:

1. Ένα σύστημα αναζήτησης λέξεων:
    a. Με χρήση ανεστραμμένου αρχείου
    b. Χωρίς χρήση βοηθητικής δομής
2. Ένα σύστημα αναζήτησης χωρικών δεδομένων:
    a. Με χρήση χωρικού ευρετηρίου που είναι βασισμένο σε σχάρα (grid)
    b. Χωρίς χρήση βοηθητικής δομής
3. Ένα σύστημα αναζήτησης μεικτής αναζήτησης με βάση την τοποθεσία αλλά και λέξεων:
    a. Με πρώτο πέρασμα αναζήτησης στο ανεστραμμένο αρχείο και δεύτερο πέρασμα στο χωρικό ευρετήριο
    b. Με πρώτο πέρασμα αναζήτησης στο χωρικό ευρετήριο και δεύτερο πέρασμα στο ανεστραμμένο ευρετήριο
    c. Χωρίς χρήση βοηθητικής δομής

Τέλος καλούμαστε να συγκρίνουμε τις αποδόσεις των τεχνικών του κάθε είδους αναζήτησης.

ΠΕΡΙΓΡΑΦΗ ΥΛΟΠΟΙΗΣΗΣ

ΑΝΑΖΗΤΗΣΗ ΜΕ ΛΕΞΕΙΣ-ΚΛΕΙΔΙΑ

Στο αρχείο text_search.py υλοποιώ όλες τις απαραίτητες συναρτήσεις για την αναζήτηση με λέξεις-κλειδιά και στο αρχείο part1.py στήνω ένα ντέμο που φαίνονται τα αποτελέσματα και οι αποδόσεις των δύο τεχνικών. Παρακάτω παρουσιάζω τις συναρτήσεις του αρχείου text_search.py και τον τρόπο λειτουργίας τους.

- **Generate_inverted_index:** Παράγει το ανεστραμμένο αρχείο για τις λέξεις κλειδιά των δεδομένων μας. Το ανεστραμμένο αρχείο έχει την μορφή δύο λιστών:
    - Μία αλφαβητικά ταξινομημένη λίστα με τις λέξεις κλειδιά που συναντάμε στα δεδομένα.
    - Μία λίστα από λίστες που ακολουθεί την ταξινόμηση της λίστα με τις λέξεις κλειδιά και αποθηκεύει για κάθε λέξη κλειδί τους αριθμούς των γραμμών των εγγραφών που περιέχουν αυτή την λέξη κλειδί.

- **Merge_join:** Παίρνει ως είσοδο την λίστα με τις λέξεις-κλειδιά που ψάχνουμε, την ταξινομεί και εφόσον είναι ταξινομημένο και το ανεστραμμένο αρχείο, κάνει την γνωστή merge_join αναζήτηση με τους pointers. Τέλος επιστρέφει τις θέσεις των keywords στο ανεστραμμένο ευρετήριο.
- **kwSearchIF:** Εδώ ουσιαστικά υλοποιείται η αναζήτηση με χρήση του ανεστραμμένου αρχείου. Αρχικά καλεί την merge_join για να βρει τις θέσεις των keywords στο ανεστραμμένο αρχείο. Στην συνέχεια μετατρέπει την λίστα των εστιατορίων της πρώτης λέξεις σε σετ και βρίσκει την τομή με τις υπόλοιπες λίστες των λέξεων. Τέλος επιστρέφει τις θέσεις των ταινιών που πληρούν τα κριτήρια αναζήτησης.
- **kwSearchRaw:** Εδώ η αναζήτηση γίνεται χωρίς την χρήση του ανεστραμμένου αρχείου. Απλά σκανάρει μία μία τις εγγραφές και κρατάει αυτές που πληρούν τα κριτήρια της αναζήτησης.

ΧΩΡΙΚΗ ΑΝΑΖΗΤΗΣΗ

Στο αρχείο spatial_search.py υλοποιώ όλες τις απαραίτητες συναρτήσεις για την χωρική αναζήτηση και στο αρχείο part2.py στήνω ένα ντέμο που φαίνονται τα αποτελέσματα και η αποδόσεις των δύο τεχνικών. Παρακάτω παρουσιάζω τις συναρτήσεις του αρχείου spatial_search.py και τον τρόπο λειτουργίας τους. (**ΣΗΜΕΙΩΣΗ**: τα borders τα υπολογίζω κατά το φόρτωμα των δεδομένων στο αρχείο loader.py)

- **create_grid:** Φτιάχνει ένα πλέγμα από τα όρια των κελιών σύμφωνα με τα borders των δεδομένων και το πλήθος των κελιών ανά διάσταση που εισάγει ο χρήστης (50 για την άσκηση άρα 50*50=2500 κελιά). Επιστρέφει μία λίστα με τα όρια για το άξονα x και μία λίστα με τα όρια για τον άξονα y.
- **Add_restaurants_grid:** Σε αυτή την συνάρτηση δημιουργούμε τα κελιά με βάση τα όρια που δημιουργήσαμε προηγουμένως και εισάγουμε τις εγγραφές μας σε αυτά. Για να βρούμε το κατάλληλο κελί κάνουμε την δυαδική αναζήτηση που περιγράφεται στην συνέχεια.
- **Binary_search:** Αυτή η συνάρτηση κάνει δυαδική αναζήτηση μιας μεταβλητής σε ένα πίνακα με όρια κελιών (είτε για τον x είτε για τον y άξονα). Την χρησιμοποιώ κυρίως για να δω σε ποιο κελί πέφτει μία μεταβλητή. Ουσιαστικά την καλούμε μία φορά για κάθε διάσταση για να βρούμε το κελί.
- **SpaSearchGrid:** Εδώ υλοποιείται η χωρική αναζήτηση με χρήση του χωρικού ευρετηρίου. Εδώ καλούμε 4 φορές την δυαδική αναζήτηση για να βρούμε όλα τα κελιά που περιλαμβάνει το παράθυρο αναζήτησης. Στην συνέχεια ελέγχουμε μία μία τις ταινίες που προκύπτουν και κρατάμε αυτές που ανήκουν στο παράθυρο αναζήτησης. Αυτό το βήμα το κάνω επειδή επιλέγω το ελάχιστο υποσύνολο των κελιών που περικλείει ολόκληρο το παράθυρο αναζήτησης και αυτό έχει ως αποτέλεσμα να υπάρχουν και εγγραφές που δεν ανήκουν στο παράθυρο αναζήτησης.
- **SpaSearchRaw:** Εδώ η αναζήτηση γίνεται χωρίς την χρήση του ανεστραμμένου αρχείου. Απλά σκανάρει μία μία τις εγγραφές και κρατάει αυτές που πληρούν τα κριτήρια της αναζήτησης.

ΧΩΡΟ-ΚΕΙΜΕΝΙΚΗ ΑΝΑΖΗΤΗΣΗ

Στο αρχείο spatio_textual_search.py υλοποιώ όλες τις απαραίτητες συναρτήσεις για την χωρο-κειμενική αναζήτηση και στο αρχείο part3.py στήνω ένα ντέμο που φαίνονται τα αποτελέσματα και η αποδόσεις των τριών τεχνικών. Παρακάτω παρουσιάζω τις συναρτήσεις του αρχείου spatio_textual_search.py και τον τρόπο λειτουργίας τους:

- **kwSearchIFAfterGrid:** Αυτή η συνάρτηση είναι ίδια με την συνάρτηση kwSearchIF με την μόνη διαφορά ότι παίρνει μία λίστα με τα αποτελέσματα της χωρικής αναζήτησης ώστε να φιλτράρει τα αποτελέσματα της αναζήτησης με λέξεις κλειδιά.
- **spaSearchGridAfterIF:** Αυτή η συνάρτηση είναι ίδια με την συνάρτηση spaSearchGrod με την μόνη διαφορά ότι παίρνει μία λίστα με τα αποτελέσματα της αναζήτησης με λέξεις κλειδιά ώστε να φιλτράρει τα αποτελέσματα της χωρικής αναζήτησης.
- **kwSpasearchIF:** Εδώ υλοποιείται η χωροκειμενική αναζήτηση κάνοντας σε πρώτη φάση την αναζήτηση με λέξεις κλειδιά καλώντας την kwSearchIF και σε δεύτερη φάση καλώντας την spaSearchGridAfterIF για να κάνει την χωρική αναζήτηση. Τα αποτελέσματα της kwSearchIF τροφοδοτούνται στην spaSearchGridAfterIF ώστε να φιλτραριστούν τα αποτελέσματα.
- **kwSpasearchGrid:** Εδώ υλοποιείται η χωροκειμενική αναζήτηση κάνοντας σε πρώτη φάση την χωρική αναζήτηση καλώντας την spaSearchGrid και σε δεύτερη φάση καλώντας την kwSearchIFAfterGrid για να κάνει την αναζήτηση με λέξεις κλειδιά. Τα αποτελέσματα της spaSearchGrid τροφοδοτούνται στην kwSearchIFAfterGrid ώστε να φιλτραριστούν τα αποτελέσματα.
- **kwSpasearchRaw:** Εδώ η χωρο-κειμενική αναζήτηση γίνεται χωρίς την χρήση του ανεστραμμένου αρχείου και του χωρικού ευρετηρίου. Απλά σκανάρει μία μία τις εγγραφές και κρατάει αυτές που πληρούν τα κριτήρια της αναζήτησης.

ΣΥΓΚΡΙΣΗ ΤΕΧΝΙΚΩΝ

ΑΝΑΖΗΤΗΣΗ ΜΕ ΛΕΞΕΙΣ-ΚΛΕΙΔΙΑ

Η αναζήτηση με χρήση ανεστραμμένου αρχείου είναι πιο γρήγορη από την αναζήτηση χωρίς. Αυτό οφείλεται στο ότι η αναζήτηση χωρίς ανεστραμμένο αρχείο, ανεξαρτήτως εισόδου θα προσπελάσει όλα τα δεδομένα για να μας επιστρέψει το αποτέλεσμα. Ενώ η αναζήτηση με ανεστραμμένο αρχείο επισκέπτεται μόνο ένα υποσύνολο από τα δεδομένα.

QUERY	RAW	INVERTED FILE
late night, breakfast/brunch, italian	0.00298sec	0.00099sec
Late night	0.00697sec	0.00201sec
greek, bar	0.00301sec	0.0sec
late night, breakfast/brunch, greek	0.00298sec	0.0sec

Μπορούμε να παρατηρήσουμε ότι η ταχύτητα της αναζήτησης με ανεστραμμένο αρχείο εξαρτάται σε μεγάλο βαθμό από το πλήθος των εγγραφών που περιλαμβάνουν οι λέξεις που αναζητάμε. Επιπλέον

στις δύο τελευταίες περιπτώσεις ο χρόνος εκτέλεσης είναι τόσο μικρός που ο μετρητής δεν αντιλαμβάνεται την διαφορά (λογικά έχει να κάνει με την ακρίβεια που αντιλαμβάνεται).

ΧΩΡΙΚΗ ΑΝΑΖΗΤΗΣΗ

Η χωρική αναζήτηση με χρήση χωρικού ευρετηρίου φαίνεται να είναι πιο γρήγορη από την αναζήτηση χωρίς ευρετήριο. Όπως φαίνεται και παρακάτω τα αποτελέσματα δεν είναι ξεκάθαρα και μάλλον οφείλεται στο ότι το αρχείο δεν είναι αρκετά μεγάλο. Γενικά όμως η απόδοση της αναζήτησης με χρήση χωρικού ευρετηρίου φαίνεται να εξαρτάται από το πλήθος των κελιών που περιλαμβάνει το παράθυρο αναζήτησης αλλά και από το πλήθος των εγγραφών που περιλαμβάνουν αυτά τα κελιά.

QUERY	RAW	GRID
51 51.50 -1 1	0.00897sec	0.00498sec
51 51.50 -1 0	0.005sec	0.00302sec
51 53 -2 1	0.00598sec	0.00800sec
53 54 -3 -1	0.00301sec	0.0sec

ΧΩΡΟ-ΚΕΙΜΕΝΙΚΗ ΑΝΑΖΗΤΗΣΗ

Για το πλήθος των δεδομένων που έχουμε η raw αναζήτηση είναι σχεδόν σε όλες τις περιπτώσεις αποδοτικότερη από τις άλλες δύο.

QUERY	RAW	IF	GRID
51 51.50 0 1 breakfast/brunch	0.00199sec	0.0179sec	0.00299sec
51.5 54 -1 1 breakfast/brunch	0.00299sec	0.20644sec	0.09078sec
51.5 54 -1 1 breakfast/brunch greek	0.00199sec	0.00302sec	0.09473sec
51 51.5 -0.5 0 breakfast/brunch greek	0.00199sec	0.00301sec	0.05586sec

Για τις υπόλοιπες τεχνικές μπορούμε να διακρίνουμε ένα μοτίβο για τις περιπτώσεις που είναι αποδοτικότερη η κάθε μία από αυτές. Στις περιπτώσεις που το παράθυρο αναζήτησης είναι μικρό αυτό συνήθως σημαίνει ότι περιορίζει κατά πολύ τα αποτελέσματα της αναζήτησης οπότε είναι αποδοτικότερο να χρησιμοποιήσουμε την kwSpaSearchGrid. Στις περιπτώσεις που οι λέξεις κλειδιά είναι αρκετά περιοριστικές είναι προτιμότερο να χρησιμοποιήσουμε την kwSpaSearchIF. Καταλήγουμε δηλαδή στο συμπέρασμα ότι είναι προτιμότερο να γίνει πρώτα η αναζήτηση που θα μας κόψει περισσότερο τα δεδομένα.