ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΜΥΕ041 - ΠΛΕ081: Διαχείριση Σύνθετων Δεδομένων

(ΕΑΡΙΝΟ ΕΞΑΜΗΝΟ 2022-23)

ΕΡΓΑΣΙΑ 3 - Top-k queries

Προθεσμία: 29 Μαΐου 2023, 11:59μμ

Στόχος της εργασίας είναι η ανάπτυξη αλγορίθμου για ερωτήσεις κορυφαίων κ (top-k queries), οι οποίες συναθροίζουν ατομικά σκορ αντικειμένων από **τρεις** πηγές. Τα αντικείμενα μπορεί να είναι εστιατόρια, η κάθε πηγή ένας ιστότοπος κρίσεων εστιατορίων (π.χ. google, TripAdvisor, yelp), και το ατομικό σκορ ενός αντικειμένου σε μία πηγή ο μέσος όρος των βαθμολογιών (ratings) που πήρε το εστιατόριο στο συγκεκριμένο ιστότοπο.

Οι δύο πρώτες πηγές seq1 και seq2 επιτρέπουν **μόνο σειριακές** προσπελάσεις (sequential/serial accesses). Στις πηγές seq1 και seq2 τα αντικείμενα είναι ταξινομημένα σε φθίνουσα σειρά του ατομικού τους σκορ εκεί. Η τρίτη πηγή rnd, επιτρέπει **μόνο τυχαίες** προσπελάσεις. Στην πηγή rnd τα αντικείμενα είναι ταξινομημένα με βάση το ID τους. Τα δεδομένα των πηγών βρίσκονται στα αρχεία seq1.txt, seq2.txt, και rnd.txt, αντίστοιχα. Σημειώστε ότι το ελάχιστο δυνατό ατομικό σκορ ενός αντικειμένου σε μία πηγή είναι 0.0 και το μέγιστο δυνατό 5.0. Σε κάθε γραμμή ενός αρχείου, ο πρώτος αριθμός είναι το αναγνωριστικό (ID) του αντικειμένου και ο δεύτερος αριθμός το ατομικό σκορ στη συγκεκριμένη πηγή. Στόχος των ερωτήσεων που θέλουμε να αποτιμήσουμε είναι η εύρεση των κορυφαίων εστιατορίων με βάση το άθροισμα των βαθμολογιών τους στις τρεις πηγές.

Καλείστε να γράψετε ένα πρόγραμμα, το οποίο παίρνει σαν όρισμα **από τη γραμμή διαταγών** (command-line argument) έναν θετικό ακέραιο k και υπολογίζει και τυπώνει **τα top-k αντικείμενα** από τις τρεις πηγές, χρησιμοποιώντας τη **συνάρτηση άθροισμα (sum)**. Για παράδειγμα το top-1 αντικείμενο είναι το 50905 με συνολικό σκορ 14.84.

Το πρόγραμμά σας θα πρέπει πρώτα να διαβάσει **εξ ολοκλήρου** το αρχείο rnd.txt και να βάλει τα σκορ των αντικειμένων σε μία δομή κύριας μνήμης (array) R, έτσι ώστε η θέση x του πίνακα να περιέχει το ατομικό σκορ του αντικειμένου με ID=x. Για παράδειγμα, R[0] = 2.07, R[1] = 2.33, κλπ. Κατόπιν, το πρόγραμμα θα πρέπει να διαβάζει γραμμές από τα seq1.txt και seq2.txt εναλλάξ (round-robin). **Προσοχή:** δεν πρέπει να διαβάσετε εξ ολοκλήρου τα αρχεία seq1.txt και seq2.txt σε δομές της κύριας μνήμης όπως πίνακες ή λίστες. Για κάθε αντικείμενο με ID=x το οποίο διαβάζουμε σειριακά από το seq1.txt ή seq2.txt:

- Αν το αντικείμενο x δεν το έχουμε δει ξανά, τότε αρχικοποιούμε το **κάτω όριο του συνολικού σκορ** του αντικειμένου να είναι το σκορ που διαβάσαμε σειριακά **συν** το score R[x] (θεωρούμε ότι έχει γίνει ένα random access στον πίνακα R για να πάρουμε το R[x]).
- Αν έχουμε ξαναδεί το αντικείμενο x, αυτό σημαίνει ότι έχουμε ήδη ένα κάτω όριο του x. Σε αυτή την περίπτωση προσθέτουμε στο κάτω όριο το ατομικό σκορ που μόλις διαβάσαμε, ώστε να προκύψει το **πλήρες σκορ** του x (γιατί τώρα έχουμε δει το x και στις τρεις πηγές).

Για παράδειγμα, όταν διαβάζουμε το αντικείμενο 33136, δηλ. το πρώτο αντικείμενο από το seq1.txt, και το σκορ του 5.00 στην πηγή seq1, προσπελαύνουμε το R[33136]=4.40 και σχηματίζουμε το κάτω όριο 9.40 για το αντικείμενο 33136. Αργότερα, όταν προσπελάσουμε ξανά το 33136 σειριακά από το seq2.txt, θα πάρουμε το σκορ του (4.28) στην πηγή seq2, το οποίο θα προστεθεί στο 9.40, ώστε να προκύψει το συνολικό σκορ του 33136, το οποίο είναι 13.68.

Μόλις έχετε δει ακριβώς k αντικείμενα, αρχικοποιήστε ένα min-heap W_k, η οποία αποθηκεύει τα top-k objects μέχρι στιγμής με βάση τα κάτω όρια ή ακριβή σκορ τους. Η κορυφή του W_k πρέπει να έχει το μικρότερο lower bound score από τα top-k αντικείμενα μέχρι στιγμής.

Από κει και πέρα συνεχίστε την εναλλάξ προσπέλαση των seq1.txt και seq2.txt και, μετά από κάθε προσπέλαση και ενημέρωση κάποιου κάτω ορίου ή συνολικού σκορ ενημερώστε το Threshold T = τελευταίο σκορ που έχουμε δει στο seq1.txt + τελευταίο σκορ που έχουμε δει στο seq2.txt + 5.0. Το T είναι το καλύτερο δυνατό σκορ οποιουδήποτε αντικειμένου δεν έχουμε δει μέχρι τώρα. Ενόσω το σκορ του κορυφαίου αντικειμένου στο W_k είναι μικρότερο του T δεν μπορούμε να τερματίσουμε και πρέπει να κάνουμε επιπλέον σειριακές προσπελάσεις στα seq1.txt και seq2.txt. Από τη στιγμή που το T γίνεται μικρότερο ή ίσο του κορυφαίου στοιχείου του W_k, είναι δυνατόν να τερματίσουμε. Έτσι, σε αυτή την περίπτωση πρέπει να ελέγξετε αν υπάρχει κάποιο αντικείμενο x που έχουμε δει ήδη αλλά δεν είναι στο W_k, το οποίο έχει πάνω όριο μεγαλύτερο από αυτό του top αντικειμένου στο W_k. Αν υπάρχει τέτοιο αντικείμενο, συνεχίζουμε τις σειριακές προσπελάσεις, αλλιώς τερματίζουμε και τυπώνουμε το W_k ως το τελικό αποτέλεσμα. Το πάνω όριο ενός αντικειμένου x μπορεί να υπολογιστεί προσθέτοντας στο κάτω όριο του x (που γνωρίζουμε ήδη) το τελευταίο σκορ που διαβάσαμε σειριακά στην πηγή όπου δεν έχουμε δει το x ακόμη.

Το πρόγραμμά σας πρέπει να δίνει στην έξοδο:

- Το συνολικό αριθμό σειριακών προσπελάσεων στα seq1.txt και seq2.txt (δηλαδή τον αριθμό των γραμμών που έχουν διαβαστεί από αυτά τα αρχεία). Το πρόγραμμά σας θα πρέπει να ελαχιστοποιεί τον αριθμό των γραμμών που έχουν διαβαστεί από τα seq1.txt και seq2.txt.
- Τα top-k αντικείμενα και τα συνολικά σκορ τους σε φθίνουσα σειρά με βάση το συνολικό σκορ.

Παράδειγμα για k=5

```
Number of sequential accesses= 3018
Top k objects:
50905: 14.84
85861: 14.76
22652: 14.74
75232: 14.74
20132: 14.74
```

Μπορείτε να χρησιμοποιήσετε έτοιμες δομές (π.χ. heap) από τη γλώσσα προγραμματισμού. Για να ελέγξετε την ορθότητα του προγράμματός σας συνίσταται να υλοποιήσετε και μία απλή μέθοδο (brute-force), η οποία διαβάζει όλα τα δεδομένα και υπολογίζει τα συνολικά σκορ από όλα τα αντικείμενα και βρίσκει τα κορυφαία k.

Παραδοτέα.

Βάλτε σέ ένα zip αρχείο τα προγράμματά σας και ένα PDF αρχείο, το οποίο θα περιέχει πληροφορίες και οδηγίες εκτέλεσης για τα προγράμματά σας. Υποβάλετε το zip αρχείο σας μέσω turnin στο assignment3@mye041

Οδηγίες για τις υποβολές:

1) Αν χρησιμοποιήσετε Java, το πρόγραμμά σας θα πρέπει να γίνεται compile και να τρέχει και εκτός Eclipse στους υπολογιστές του εργαστηρίου. **Μην χρησιμοποιείτε packages**.
2) Αν χρησιμοποιήσετε Python, μην χρησιμοποιήστε έτοιμες υλοποιήσεις αλγορίθμων από βιβλιοθήκες (π.χ. pandas, numpy) και μην υποβάλετε κώδικα για interactive programming (π.χ. ipython).
3) Υποβάλετε τις εργασίες σας σε ένα **zip** αρχείο (**όχι rar**) το οποίο πρέπει να περιλαμβάνει όλους τους κώδικες καθώς και ένα αρχείο τεκμηρίωσης το οποίο να περιγράφει τη μεθοδολογία σας και να περιλαμβάνει το PDF αρχείο. **Μην υποβάλετε αρχεία δεδομένων**.
4) Μην ξεχνάτε να βάζετε το όνομά σας (σε greeklish) και το ΑΜ σε κάθε αρχείο που υποβάλετε.
5) Ο έλεγχος των προγραμμάτων σας μπορεί να γίνει σε άλλα αρχεία εισόδου από αυτά που σας δίνονται, άρα θα πρέπει ο κώδικάς σας να μην εξαρτάται από τα συγκεκριμένα αρχεία εισόδου που σας δίνονται.