Ανάπτυξη Λογισμικού για Πληροφοριακά Συστήματα

Part 1

Οργάνωση αρχείων:

- files-creation:
  file_creation.c δημιουργεί sample αρχείου με τυχαίους αριθμούς σε συγκεκριμένο range. Υπάρχει makefile και εντολή εκτέλεσης της μορφής
  ./create filename num_of_records min max
  1.txt 2.txt δύο sample αρχείων
- src:
  main.c εκτελεί-τεστάρει την συνάρτηση RHJ
  RadixHashJoin.c υλοποιεί την RHJ
  Τα υπόλοιπα αρχεία είναι βοηθητικά.Υπάρχει makefile και η εκτέλεση της main είναι της μορφής
  ./run file1 file2
  π.χ. ./run ../files-creation/1.txt ../files-creation/2.txt > res.txt

Τρόποι υλοποίησης RHJ:

- Αρχικά δημιουργούνται οι δύο σχέσεις (relation) όπου διαβάζεται το κάθε αρχείο και αποθηκεύνται τα απαραίτητα στοιχεία payload και key.Τα key αρχίζουν από 1.
- Έπειτα επιστρέφεται στην μεταβλητή Result το αποτέλεσμα της συνάρτησης RadixHashJoin.
- Τέλος αυτό εκτυπώνεται για επαλήθευση του αποτελέσματος

- Η συνάρτηση FirstHash (n τελευταία bits) δημιουργεί το ιστόγραμμα και επιστέφει το νέο relation οργανωμένο σε buckets

- SecondHash k bit αριστερά μετά τα πρώτα n

- Συνολικά διατρέχονται όλοι οι κάδοι και για αυτόν με μικρότερο μέγεθος εγγραφών δημιουργείται η δομή HashBucket που περιέχει δύο πίνακες chain και bucket

- Συγκεκριμένα αυτοί αρχικοποιούνται με -1 και έτσι αν στον buket περιεχόμενα με αυτή την τιμή δηλώνουν άδεια ενώ στον chain όσες θέσεις έχουν -1 δείχνουυ ν το τέλος της αλυσίδας.Χρησιμοποιήθηκε το -1 και όχι το 0 για την δήλωση του τέλους

- Η ScanBuckets λαμβάνοντας υπ'όψιν από την μία σχέση όλο το bucket και από την άλλη την δομή HashBucket, διατρέχονται τα στοιχεία του bucket και για καθένα από αυτά χρησιμοποιώντας την τιμή επιστροφής της HashFunction διατρέχεται η αλυσίδα για σύγκριση των payload.Αν είναι ίδια αποθευκεύονται τα keys στην δομή Result

- Η δομή αυτή είναι ένα struct με το μέγεθος της λίστας και με δείκτη στην αρχή και το τέλος της λίστας κόμβων τύπου result_node

Περισσότερα σχόλια μέσα στα αρχεία

Note:

Στο προηγούμενο commit “previous version” η έκδοση διαφέρει από αυτήν στην αρχικοποίηση της δομής bucket του ευρετηρίου αλλά δεν είναι τελική γιατί είναι πιο αργή σε χρόνο απο αυτήν.
Στην σύγκριση των κάδων είναι αναγκαία η αρχικοποίηση της δομής bucket με -1.
Σκεφτήκαμε ότι η μη αρχικοποίηση θα μπορούσε να απογευχθεί με τον εξής τρόπο:
Στην δομή HashBucket κράταμε για το bucket ένα πίνακα από struct bucket(int loop,value).
Έτσι περνώντας στις SecondHash τον αριθμό της επανάληψης (....,i) συγκρίνουμε το τρέχον loop με το παλιό

Φοιτητές:
Καλογερόπουλος Ιωάννης 1115201500057
Κότσι Ηρακλή 1115201500073
Παπασωτηρίου Ηλίας 1115201500123