Ανάπτυξη Λογισμικού για Υπολογιστικά Συστήματα (PROJECT)

Μέρος Πρώτο: Τελεστής Radix Hash Join
Μέρος Δεύτερο: Σύζευξη και εφαρμογή φίλτρου σε επερωτήσεις

Φοιτητές:

Ονοματεπώνυμο: Κατσορίδας Ιωάννης
Αριθμός Μητρώου: 1115201400066
SDI: sdi1400066

Ονοματεπώνυμο: Μητράκης Γεώργιος
Αριθμός Μητρώου: 1115201400107
SDI: sdi1400107

ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ 2018-2019

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΜΕΡΟΣ ΠΡΩΤΟ:

Εκτελέσιμα:

./DataGenerator <n>
όπου:
n = 1 για τη δημιουργία relations με increasing payloads (== 0,1,2... REL_SIZE),
n = 2 για τη δημιουργία relations με identical payloads (== 1),
n = 3 για τη δημιουργία relations με exclusive per relation payloads (R -> 1 και S -> 2),
n = οτιδήποτε άλλο για τη δημιουργία relations με random payloads

./Caramel
για την δημιουργία των relations και την πράξη του join.

./UnitTests
για το unit testing των συναρτήσεων

Μεταγλώττιση:

make

Αρχεία και περιεχόμενά τους:

- DataGenerator.c, DataGenerator.h

Το πηγαίο και το αρχείο βιβλιοθήκης του, από τα οποία προκύπτει το εκτελέσιμο που δημιουργεί 2 αρχεία με ονόματα “DataRelationR.txt” και “DataRelationS.txt” με τυχαία payloads. Το εκτελέσιμο αυτό εκτελείται ανεξάρτητα από τα υπόλοιπα κομμάτια και εκτελείται μόνο αν θέλει ο χρήστης να αλλάξει τα δεδομένα με τα οποία τρέχει τον αλγόριθμο.

- DataParse.c, DataParse.h

Το πηγαίο και το αρχείο βιβλιοθήκης του που περιλαμβάνει τις συναρτήσεις για δημιουργία relation από τα παραπάνω txt files, δημιουργία ιστογράμματος, αθροιστικού ιστογράμματος και δημιουργία νέου κατακερματισμένου relation από ένα ήδη υπάρχον, όπως αυτά περιγράφονται στην εκφώνηση

- Globals.h

Το κύριο header file, που περιλαμβάνει ορισμούς global δομών συναρτήσεων και μεταβλητών και γίνεται included σε όλα τα πηγαία.

- main.c

Περιλαμβάνει την main συνάρτηση του εκτελέσιμου, η οποία δημιουργεί δύο relations R και S, κάνει Radix Hash Join , αποθηκεύει το αποτέλεσμα του join αυτού και επιστρέφει.

- RadixHashJoin.c

Το πηγαίο που κάνει Join δύο relations. Δημιουργεί ιστόγραμμα και αθροιστικό ιστόγραμμα για κάθε relation, τα κατακερματίζει δημιουργώντας δύο νέα relations και κάνει αναζήτηση για ίσα payloads μεταξύ στοιχείων του R και του S που ανήκουν στο ίδιο bucket. Δημιουργείται ευρετήριο για το bucket του relation με τα λιγότερα tuples και για κάθε tuple του R που είναι ίσο με κάποιο tuple του S στο ίδιο bucket, προστίθενται τα keys στη λίστα αποτελεσμάτων, η οποία και επιστρέφεται.

- Results.c, Results.h

Περιέχουν συναρτήσεις για διαχείριση της λίστας αποτελεσμάτων του Join.

- Index.c, Index.h

Περιέχουν τη δομή του index{bucket array, chain array} και συναρτήσεις που διαχειρίζονται τη δημιουργία, την αρχικοποίηση, την αναζήτηση στον εκάστοτε κάδο και τη διαγραφή του.

UnitTests

Όλες οι συναρτήσεις που υλοποιήθηκαν δοκιμάζονται για την αποτελεσματικότητά τους μέσω unit test. Τα test αυτά υλοποιήθηκαν με τη χρήση του Framework: CUnit και βρίσκονται στο αρχείο UnitTests.c. Ως είσοδο παίρνουν καρφωτά συγκεκριμένες τιμές και ελέγχεται η έξοδος από τις διάφορες συναρτήσεις ως προς την εγκυρότητά τους με βάση την ομοιότητά τους με τα αναμενόμενα αποτελέσματα.

ΜΕΡΟΣ ΔΕΥΤΕΡΟ:

Αποθήκευση Σχέσεων Στη Μνήμη:

Το πρόγραμμα δέχεται ως είσοδο μια σειρά από ονόματα αρχείων, κάθε ένα από τα οποία περιέχει σε δυαδική μορφή έναν πίνακα με τις τιμές του. Χρησιμοποιώντας τη συνάρτηση loadRelation του αρχείου Relation.cpp του πακέτου submission του διαγωνισμού SIGMOD (κατάλληλα τροποποιημένου ώστε να υποστηρίζει τη γλώσσα C στην οποία είναι γραμμένος ο υπόλοιπος κώδικας αντί για τη C++ που είναι το πακέτο submission), φορτώνονται στη μνήμη όλες οι σχέσεις που έχουν δοθεί σαν input. Οι σχέσεις φορτώνονται σε μια δομή τύπου: table** όπου κρατώνται τόσο ο αριθμός των στηλών και των κολόνων, όσο και οι ίδιες οι τιμές του πίνακα.

Έπειτα, γεμίζει μια δομή Metadata* για κάθε κολόνα του πίνακα, η οποία περιέχει στατιστικά όπως ο αριθμός των μοναδικών τιμών (distincts) ανά κολόνα ή η μέγιστη και η ελάχιστη τιμή. Αυτά τα στατιστικά θα χρησιμοποιηθούν για την επιλογή της ένωσης ή σύγκρισης που θα πραγματοποιηθεί πρώτα σε ένα query, ανάλογα με το ποιο αναμένεται ότι θα επιστρέψει τα λιγότερα αποτελέσματα.

Επεξεργασία των Queries:

Μετά την αποθήκευση των σχέσεων στη μνήμη, το πρόγραμμα αναμένει να εισέλθουν ως είσοδος κάποιες επερωτήσεις, από τις οποίες θα προκύψουν τα επιθυμητά αποτελέσματα. Διαβάζει τις επερωτήσεις ανά ομάδα, όπου η κάθε ομάδα τελειώνει όταν διαβαστεί ο χαρακτήρας “F”. Μετά το διάβασμα μιας ομάδας, εκτελεί το κάθε query με τη σειρά εισόδου του και εκτυπώνει το άθροισμα των τιμών που θα ήταν το αποτέλεσμα της επερώτησης. Έπειτα, περιμένει εκ νέου να δοθούν νέα queries.

Για την επεξεργασία των επερωτήσεων, χρησιμοποιείται η δομή Query η οποία περιέχει δείκτες σε δομές όπως Column_t, Comparison_t, Query_relation_t οι οποίες περιέχουν τις απαραίτητες πληροφορίες σχετικά με την επερώτηση. Έπειτα, η δομή αυτή δίνεται ως είσοδος στη συνάρτηση ExecuteQueries η οποία ανάλογα με τον τύπο του, πραγματοποιεί αντίστοιχες πράξεις.

Εκτέλεση Κατηγορημάτων:

Προτού εκτελεστούν τα κατηγορήματα, δίνεται σε κάθε ένα από αυτά ένας βαθμός προτεραιότητας. Αυτός ο βαθμός προκύπτει από τον αναμενόμενο αριθμό αποτελεσμάτων που θα επιστρέψει αυτό το κατηγόρημα, σε περίπτωση που τα στοιχεία ήταν ομοιόμορφα κατανεμημένα. Το πραγματοποιεί αυτό χρησιμοποιώντας τα στοιχεία που σύλλεξε μετά την αποθήκευση των σχέσεων. Όσο λιγότερα αποτελέσματα αναμένεται να επιστραφούν, τόσο πι γρήγορα θα εκτελεστεί το επερώτημα. Αυτό γίνεται καθώς, με δεδομένο πως όπως και να εκτελεστούν θα επιστρέψουν τα ίδια αποτελέσματα, αν από την αρχή υπάρχει μικρός αριθμός αποτελεσμάτων, τα υπόλοιπα επερωτήματα θα εκτελεστούν πιο γρήγορα.

Για τη διαδοχική εκτέλεση των κατηγορημάτων, χρησιμοποιείται μια συνδεδεμένη λίστα, κάθε κόμβος της οποίας περιέχει έναν πίνακα από ενδιάμεσα αποτελέσματα. Κάθε φορά που εκτελείται ένα predicate, θα υπάρχει (αν τα αποτελέσματα δεν είναι μηδέν) ένας αριθμός από σειρές οι οποίες ικανοποιούν τη σχέση που έχει δοθεί. Αυτές οι σειρές αποθηκεύονται σε έναν ενδιάμεσο πίνακα αποτελεσμάτων, ώστε την επόμενη φορά που μία από τις σχέσεις, που βρίσκονται σε κάποιο πίνακα, βρεθεί σε άλλο ένα κατηγόρημα, να χρησιμοποιηθούν οι τιμές που βρίσκονται στα ενδιάμεσα αποτελέσματα και όχι κάποιες άλλες οι οποίες έχουν ήδη αποκλειστεί λόγω κάποιας προηγούμενης συνθήκης. Στη περίπτωση που ένα κατηγόρημα περιλαμβάνει σχέση ή σχέσεις των οποίων καμία δεν βρίσκεται σε κάποιο ενδιάμεσο αποτέλεσμα, δημιουργείται ένας νέος πίνακας ενδιάμεσων αποτελεσμάτων που προστίθεται στη λίστα. Αν βρεθεί κατηγόρημα το οποίο περιλαμβάνει σχέσεις από 2 ξεχωριστούς πίνακες, τότε αυτοί ενώνονται σε έναν.

Έτσι, ανάλογα με τη προτεραιότητα, εκτελούνται ένα-ένα όλα τα κατηγορήματα. Οι κατηγορίες στις οποίες μπορεί να ανήκει ένα κατηγόρημα είναι τρεις: να είναι σύγκριση με έναν αριθμό (>, <, =), να είναι ένωση (Join) μεταξύ κολόνων της ίδιας σχέσης και να είναι join μεταξύ ανεξάρτητων σχέσεων.

Στη πρώτη κατηγορία, ελέγχονται όλες οι τιμές της κολόνας της σχέσης (ή αντίστοιχα οι τιμές που βρίσκονται στις σειρές που υπάρχουν σε κάποιο πίνακα ενδιάμεσων αποτελεσμάτων, αν έχει ήδη πραγματοποιηθεί κάποια πράξη στην ίδια σχέση) και αυτές που ικανοποιούν τη συνθήκη, αποθηκεύονται σε ένα πίνακα αποτελεσμάτων.

Στη δεύτερη κατηγορία, σε αντίθεση με τις άλλες ενώσεις, δεν

εκτελείται ο αλγόριθμος του Radix Hash Join, καθώς πρόκειται για κολόνες τις ίδιας σχέσης. Αντίθετα, ελέγχονται όλες οι σειρές που βρίσκονται είτε σε κάποιο ενδιάμεσο αποτέλεσμα, είτε από τον αρχικό πίνακα αν δεν έχει προηγηθεί κατηγόρημα στη συγκεκριμένη σχέση, ώστε να δοθούν σαν αποτελέσματα μόνο οι σειρές της σχέσης, των οποίων οι εν λόγω κολόνες έχουν ίδια τιμή. Κάτι παρόμοιο γίνεται και στη περίπτωση που δύο σχέσεις βρίσκονται στον ίδιο πίνακα ενδιάμεσων αποτελεσμάτων και φτάνει ένα κατηγόρημα για αυτές.

Στη τελευταία περίπτωση πρέπει να εκτελεστεί ο αλγόριθμος Radix Hash Join που υλοποιήθηκε στο προηγούμενο μέρος. Ο αλγόριθμος παίρνει σαν είσοδο δύο σχέσεις και επιστρέφει τις σειρές που έχουν ίδια τιμή. Για να χρησιμοποιηθεί, λοιπόν, ο RHJ, πρώτα φτιάχνονται σχέσεις από τις κολόνες που ενώνονται. Αν κάποια υπάρχει σε κάποιο ενδιάμεσο αποτέλεσμα, τότε ο αριθμός της εκάστοτε σειράς δίνεται από την θέση στον ενδιάμεσο πίνακα. Αν καμία από τις δοθείσες σχέσεις δε βρίσκεται σε κάποιο ενδιάμεσο αποτέλεσμα, τότε δημιουργείται ένας νέος πίνακας και προστίθεται στη λίστα. Αν από την άλλη υπάρχει μόνο μία από τις δύο σχέσεις, τότε ο πίνακας στον οποίο βρίσκεται η άλλη μεγαλώνει κατά μία στήλη, και προστίθενται εκεί οι τιμές της νέας σχέσης. Αν από την άλλη και οι δύο σχέσεις βρίσκονται σε διαφορετικά ενδιάμεσα αποτελέσματα, τότε, όπως έχει ήδη αναφερθεί, οι δύο πίνακες ενώνονται σε έναν κοινό.

Εκτύπωση Αποτελεσμάτων:

Αφού εκτελεστούν όλα τα κατηγορήματα, θα προκύψει μια σειρά από πίνακες ενδιάμεσων αποτελεσμάτων. Σε εκείνο τη σημείο λαμβάνονται υπόψη οι επιθυμητές κολόνες προς εκτύπωση. Αν και οι δύο κολόνες βρίσκονται στον ίδιο πίνακα, τότε από αυτόν εκτυπώνονται οι κολόνες ως έχουν. Αν από την άλλη οι κολόνες βρίσκονται σε πίνακες ενδιάμεσων αποτελεσμάτων που δεν είναι ενωμένοι, τότε δημιουργείται ένας νέος πίνακας ενδιάμεσων αποτελεσμάτων που αποτελεί το καρτεσιανό γινόμενο των δύο προηγούμενων. Και από αυτόν τον πίνακα, εκτυπώνονται οι επιθυμητές κολόνες.