CSCI 540 - Advanced Databases Course Project Progress Report for December 7th

Larry Lynn
Liessman Sturlaugson
Cole Schock

December 6, 2011

Abstract

The iMinMax team added the remaining required performance metrics to their algorithms, namely, single query time, average query time, and tree segments. The team also adapted sequential scan algorithms for the three query types (point, range, and k-nearest neighbor) for comparing to iMinMax.

1 Code Progress

Having implemented the algorithms for the three required queries, the iMinMax team proceeded to incorporate the following: a process to output the tree segments (interval covered by each leaf), sequential scan algorithms for comparison to iMinMax, timers for individual queries (as well as code to find average query time over a set of queries), and code to parse and execute query files passed from the command line.

1.1 Tree Segments

Because the variables associated with the leaves are private to the *btree* class of the B$^+$-tree code [1], determining the tree segments required changing the source code directly (as with the counts of nodes accessed during a given query). The code to output the tree segments was added as a single public method to the *btree* class in btree.h. A wrapper method to call this base class method was then added to the derived class *btree_multimap* in btree_multimap.h, as this derived class is the one used for our project.

1.2 Sequential Scan

Sequential scan methods were added for each of the three query types. These methods were adapted from the code of the iDistance team [2]. Some conversion was required because of the slightly different data structures being used by the two teams.

1.3 Query Timing

The query timing code was added to main(), as this is the method responsible for calling the various queries. When timing multiple queries, the code maintains a vector of times, which it uses to find the average query time once all the queries in the query file have been executed.

1.4 Query File Parsing

The existing CSV parsing code was adapted to also support query file parsing. The method returns a vector of points. For point queries, each entry in the vector is the point of a point query. For range queries, consecutive pairs of entries define the “upper” and “lower” corner points for a range query. For KNN queries, each entry in the vector is the point at which to perform the query. The k parameter for these KNN queries is passed via the command line per the project instructions.

1.5 Code Debugging

The team is currently debugging the algorithms using the provided datasets and the provided queries.

2 Progress on the Final Paper

The team expanded the introduction and the description of iMinMax for range and KNN queries.

They also added the section describing the datasets, enumerating the varying data distributions, number of data points, and number of dimensions.

References

[1] Timo Bingmann. STX B+ Tree C++ Template Classes, November 2011, https://idlebox.net/2007/stx-btree/.

[2] Timothy Wylie, Michael Schuh, and Melissa Dale. An implementation of the iDistance multidimensional indexing method with extensions, December 2011, http://code.google.com/p/idistance/.