Nearest Neighbor Queries

Nick Roussopoulos, Stephen Kelley and Frederic Vincent

Abhishek Srivastava
Student ID: 861307778

February 16, 2017
CS 236, Winter 2017

The problem:
The paper notes that finding nearest neighbor queries are quite common for the spatial datasets but processing them requires very different approach than for location or range queries. This kind of queries can be generalized for different purposes which make them more important to be processed effectively and can be modified to serve multiple requirements.

The contribution:
The authors propose an extension of *nearest neighbor principle* on R-tree and utilizes the branch and bound traversal to find the k-nearest neighbors objects from a query point. The algorithm considers a heuristics to prune the branches and reduces the search space for the nearest objects. It first traverses top-down exploring subtrees containing query points and then traversing upwards to other subtrees which can contains nearest neighbors.

The method:
The Authors proposes two metrics for the heuristics which can be used to prune the subtrees of the search space in R-tree:

- MINDIST: Based on the minimum distance between an object and a point. If query point is inside the bounding region distance is zero. If not, distance between the query point and the nearest edge of bounding region from the point is considered.
- MINMAXDIST: This provides us the upper bound of the distance to any object inside a bounding region. This prunes the regions which have MINDIST higher than the upper bound. This guarantees that there is no nearest object between the query point and MINMAXDIST.

Not only distance from query point provides an optimal bounding region but it is also dependent on the size and layout of the objects as well. Authors provided 3 methods of pruning using MINDIST and MINMAXDIST which are used at different time during traversal. Authors implemented an ordered depth first traversal, and ordering metric is calculated and sorted into branch lists to prune unnecessary branches as it descends down the tree. Iteration are done on the active branch lists with different pruning strategies to find nearest neighbors.

This algorithm provided by Authors consider 1-NN but can be easily extended to the find k nearest neighbors with other trade-offs.

Comments:
The paper presents an extension of k-NN algorithm on the R-trees. It showed that k-NN can be scaled for the data-size and number of neighbors. They benchmarked the performance of two heuristics. MINDIST found out to be more effective.

However, it has some short comings:

- To find k-Nearest Neighbors we have to store k active branches in the main memory.
- Join operation or common neighbor queries of different points can be not efficiently.