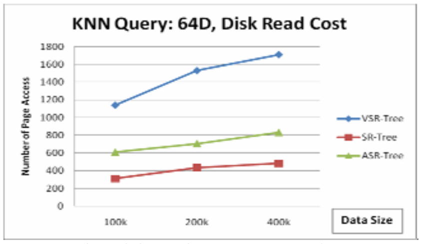


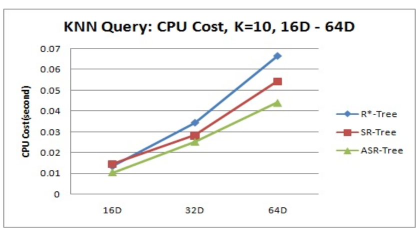
(b) Disk read cost vs. Data size

Figure 6. Effect of Data Size

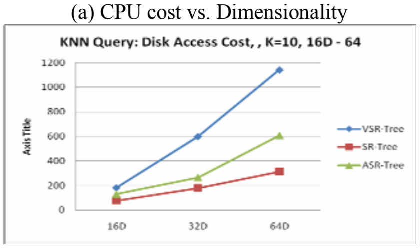
From above figures, it is evident that ASR-Tree outperforms the other two methods in terms of the response time for the three datasets. The graph shows when data size is less than 400k, the page access overhead of SR-Tree and ASR-Tree hold a big gap but their total CPU costs nearly are the same. This phenomenon is considered as the evidence that the CPU cost become the dominant component in the overall query processing cost in MBR based index structures when dimensionality increases.

C. *Effect of Dimensionality*

In our second experiment, we measure the impacts of the dimensionality in MBR based kNN search task. The detailed experimental results are presented in figure 7. As the experiments in last section, ***k*** is also set to be 10 and indexes are constructed in the optimal bulk load manner. In this experiment, data size is 100k.



(a) CPU cost vs. Dimensionality



(b) Disk read cost vs. Dimensionality

Figure 7. Effect of Dimensionality

From the experimental results, we can see that although ASR-Tree has more page access than other two approaches, it still have less response time than VSR-Tree and SR-Tree. Because of the only difference between the SR-Tree and ASR-Tree is its pruning strategies, they are same in index construction and maintenance, thus we can owe the superiority of ASR-Tree to the $\mathrm{MINMAXDIST}_{up}$ metric and the two enhanced pruning strategies ***H1***' and ***H2***'.

D. *Effectiveness of Enhanced Pruning Heuristics*

To evaluate the effectiveness of enhanced pruning heuristics ***H1'*** and ***H2'***, we devise this experiment to evaluate the ration of (visited leaves node / all the leaves nodes). The experimental settings are: ***k*** of kNN is 1000, data size is 400k and the dimensionality is up to 256. The results are plot into figure 8.

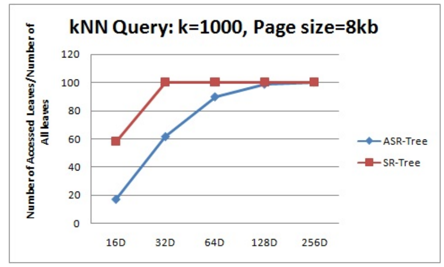


Figure 8. Effect of Enhanced Pruning Heuristics

From section C, we conclude that the differences between ASR-Tree and SR-Tree are the pruning heuristics ***H1'*** and ***H2'*** during query processing. The results are clearly shown in figure 8. When dimensional is less than 64, the effectiveness of ***H1'*** and ***H2'*** is evident, and many nodes visited in SR-Tree will be filtered in ASR-Tree. That is why ASR-Tree outperform R*-Tree and SR-Tree in the first two measurements. However, with the increase of dimensionality, the intrinsic of R*-Tree family techniques make them suffered and visit almost all the leaves node, then the performance decrease rapidly, this is the so-called *curse of dimensionality*. In this case, traditionally R-Tree based pruning techniques only aims to reduce the I/O cost may become impractical, the high-dimensional kNN search pruning techniques should consider reduce both computation cost and I/O cost to improve the overall performance.

In this paper, we propose an approach to quickly obtain a tightly approximate upper bound value of ***MINMAXDIST***, and then we use this approximate upper bound instead of its precise value to improve the search performance.

REFERENCES

[1] Christian Böhm, Stefan Berchtold, Daniel A. Keim: Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. ACM Comput. Surv. (CSUR) 2001.33(3), pp. 322-373.

[2] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, Bernhard Seeger, The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. SIGMOD 1990, pp. 322-331.

[3] Nick Roussopoulos, Stephen Kelley, Frédéic Vincent. Nearest Neighbor Queries. SIGMOD 1995, pp. 71-79.

[4] Jinchuan Chen, Reynold Cheng. Efficient Evaluation of Imprecise Location-Dependent Queries. ICDE 2007, pp. 586-595.

[5] George Beskales, Mohamed A. Soliman, Ihab F. Ilyas. Efficient search for the top-k probable nearest neighbors in uncertain databases. PVLDB, 2008, 1(1), pp. 326-339.

[6] Marios Hadjieleftheriou, George Kollios, Vassilis J. Tsotras, Dimitrios Gunopulos: Indexing spatiotemporal archives. VLDB J. (VLDB), 2006, 15(2), pp. 143-164.

[7] Yufei Tao, Dimitris Papadias, Qiongmao Shen. Continuous Nearest Neighbor Search. VLDB 2002, pp. 287-298.