I-LSH: I/O efficient c-Approximate Nearest Neighbor Search in High-dimensional Space

Want Liu[1], Hachen Wang[1], Ying Zhang[1], Wei Wang[2] and Lu Qin[1]
[1]University of Technology Sydney [2]University of New South Wales

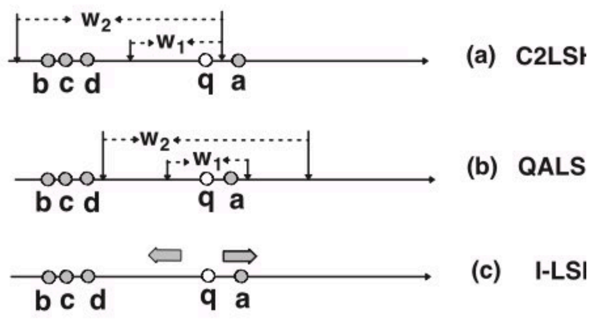
Problem Definition

Given a dataset D with n points in a d-dimensional space, denoted by R^d, the coordinate value of each object o on the ith dimension is denoted as o[i]. The c-approximate nearest neighbor is defined as follows:

- ***c-approximate nearest neighbor* (c-*ANN*)**: *For a given query object* q *and a* d-*dimensional dataset* D, *suppose* o^* *is the nearest neighbor of* q *with distance* R^*, *a* c-*approximate nearest neighbor of* q *is a data object* $o \in D$ *such that* $\| o, q \| \leq cR^*$ *where* c *is the approximate ratio.*
- $\| o1, o2 \|$ *denotes the Euclidean distance between two objects o1 and o2*

Motivation

- Existing LSH methods adopt the *bucket exponential expansion* strategy
- A c^2 approximate method requires a larger number of hash functions. It is not I/O-efficient enough.



Major contributions

- We propose a new c-approximate nearest neighbor search algorithm, namely I-LSH, which uses a natural incremental search strategy on the projected dimensions.
- We provide rigorous analysis to demonstrate the correctness and efficiency of our proposed methods.
- We perform an extensive performance evaluation against two state-of-the-art I/O efficient c-ANN algorithms regarding I/O costs and result accuracy.

Locality-Sensitive hashing family

- $h_i(o) = \vec{a}_i \cdot \vec{o}$
- $\vec{a}_i$ is a d-dimensional vector whose elements are chosen randomly following the normal distribution
- If $\| o, q \| \leq R$, $Pr[\| h(o), h(q) \| \leq \frac{w}{2}] \geq p_1$
- If $\| o, q \| \geq R$, $Pr[\| o, q \| \leq \frac{w}{2}] \leq p_2$
- w is a pre-defined parameter

Locality-Sensitive hashing family

- Let $s = \| o, q \|$, the possibility that o collides with q is:

$$p(s) = Pr[\| h(o), h(q) \| \leq \frac{w}{2}] = \frac{1}{\sqrt{2}\pi} \int_{-\infty}^{-\frac{w}{2s}} e^{\frac{-x^2}{2}}$$

- p_1 and p_2 can be calculated using the formula above

i-LSH

- A single LSH function already can keep the distance relationship with a possibility, but to boost the success possibility, multiple LSH functions are required
- There are two steps of i-LSH: (1) indexing and (2) query processing

i-LSH

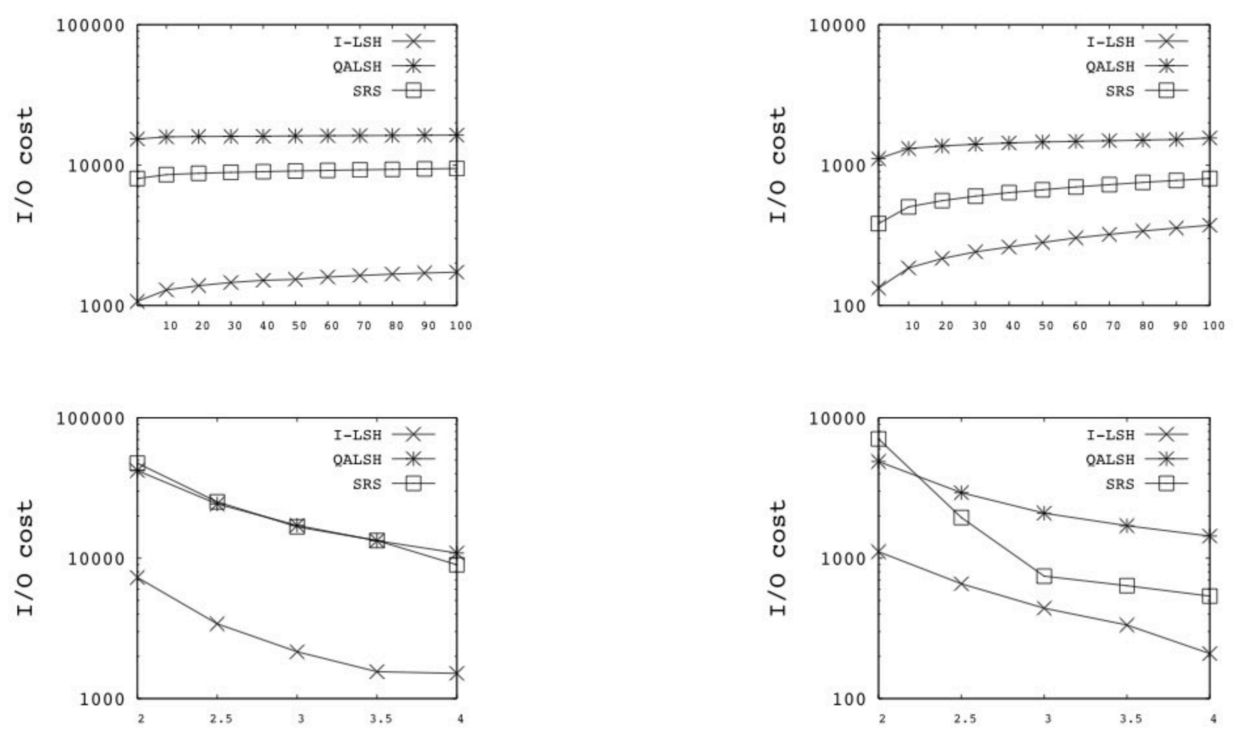
(1) Indexing step: i-LSH uses m lsh functions to project all the n data points and store the m*n hash values into m B+ trees.

(2) Query processing step: a query object q in the d-dimensional space will be mapped into m projected dimensions, then the objects and their hash values will be incrementally accessed according to their projected distances in m projected dimensions. Only a small size of candidates will be checked for their real distance.

Experimental results

- We conduct experiments over two real-world datasets: tiny and million-song compared with two c-aNN algorithms with theoretical guarantee: SRS and QALSH
- Evaluation metic: I/O cost. (1 sequence I/O is considered as 0.1 random I/O)

Experimental results



Conclusion

Under the same theoretical guarantee, I/O performance of I-LSH consistently outperforms QALSH and SRS under all settings.

Reason:

(1) I-LSH adopts a natural incremental search strategy, which can achieve c-approximate estimation (unlike the c^2 approximation of QALSH

(2) I-LSH can take advantage of efficient sequential I/O brought by the B+ tree.

VLDB 2018, Rio de Janeiro