Steiner-Hardness: A Query Hardness Measure for Graph-Based ANN Indexes

Zeyu Wang[1] Qitong Wang[2] Xiaoxing Cheng[3] Peng Wang [1]* Themis Palpanas[2] Wei Wang[1]

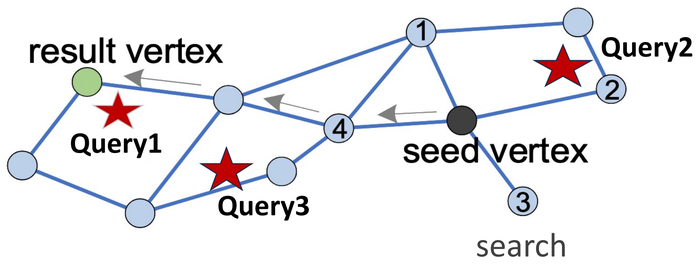
[1]Fudan University, Shanghai, China [2] Université Paris Cité, Paris, France [3]Tongji University, Shanghai, China

zeyuwang21@m.fudan.edu.cn Repo: https://github.com/CaucherWang/Steiner-hardness

1 Background

ANN Query on High-D Vectors

- Find the top-k vectors with smallest distance (L2/IP/Cosine)
- Graph Indexes are the SOTA for ng-approximate result



Query1	0.5	-0.1	0.3	0.3	0.8	...
Query2	-0.1	-0.9	0.2	0.1	0.1	...
Query3	0.8	-0.7	-0.3	-0.6	0.5	...

index

recall=96%, 0.25ms

recall=90%, 0.80ms

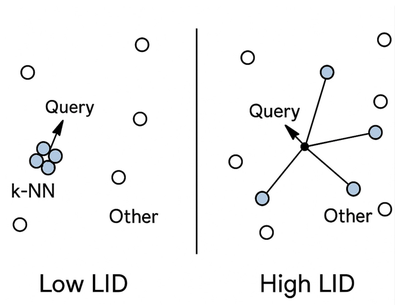
recall=40%, 7.55ms

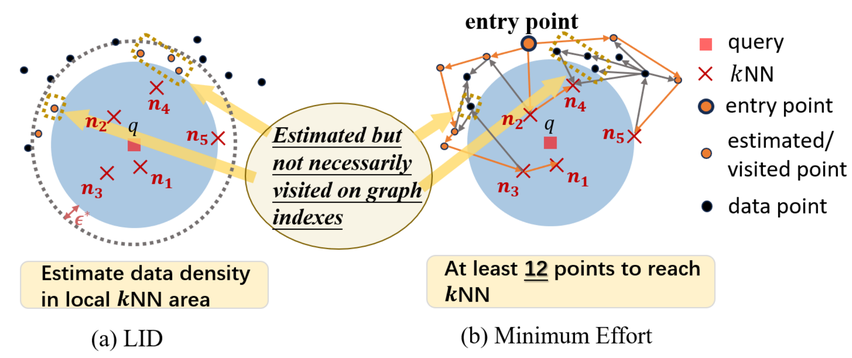
Why does the performance of the queries vary significantly?

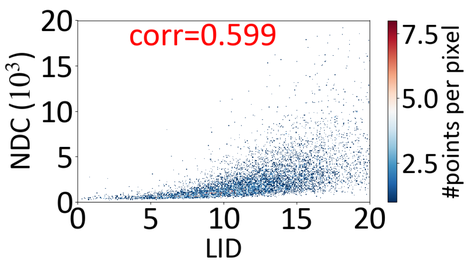
How can we define the hardness of vector ANN queries?

2 Limitation of Distance-Based Measures

$$LID(r) = \lim_{\varepsilon \to 0} \frac{\ln(F((1+\varepsilon)r)/F(r))}{\ln((1+\varepsilon)r/r)}$$





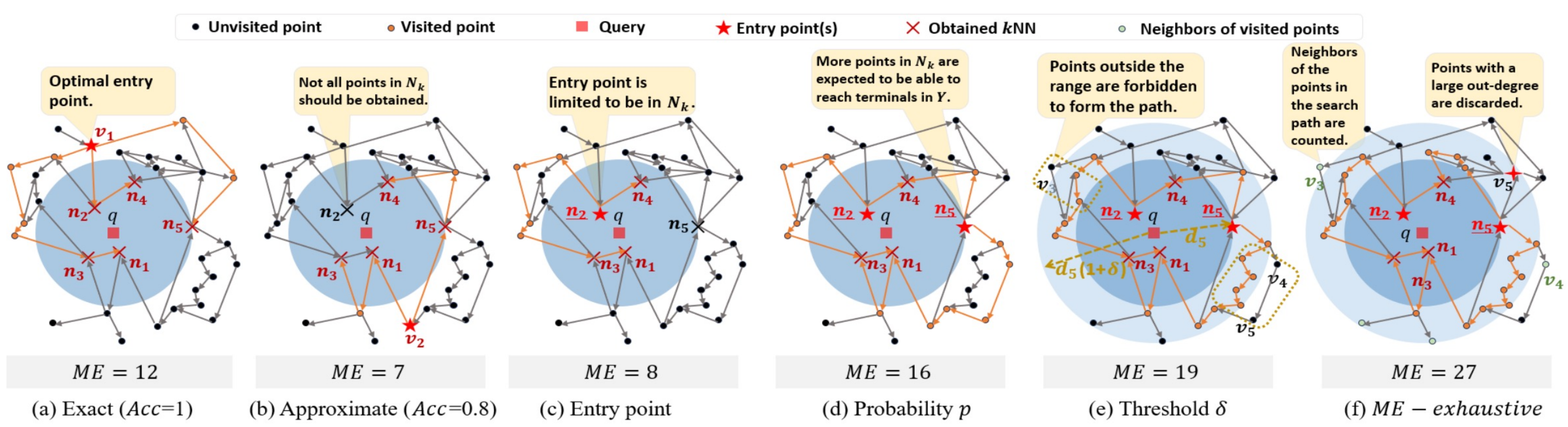


Hardness of queries is decided by graph connections to kNN.

Measures describing local data distribution (e.g. LID) are totally unaware of graph connections!

3 *Steiner*-Hardness

Minimum ANN Query Effort Analyses on A Graph



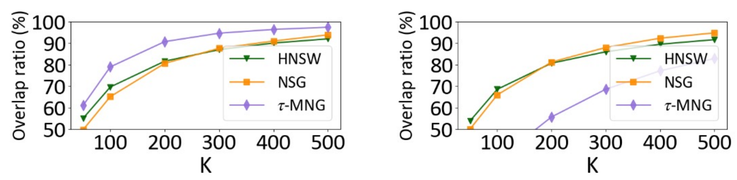
$ME = 12$	$ME = 7$	$ME = 8$	$ME = 16$	$ME = 19$	$ME = 27$
(a) Exact (Acc=1)	(b) Approximate (Acc=0.8)	(c) Entry point	(d) Probability p	(e) Threshold δ	(f) $ME - exhaustive$

- Classical Steiner-Tree problem: Given a root vertex r, a group of terminals T, find the smallest subgraph G', s.t. for any $t \in T$, there is a path from r to t on G'.
- Strict **Lower Bound** of ANN query: The size of the optimal Steiner-tree for any possible root vertex (seed vertex), where the terminals are kNN of the query.
- Constraint on the lower bound to make it closer to the real effort: (1) the root vertex is limited to one of kNN (i.e., skip Phase-1 search), (2) not all kNNs are required to be accessed (recall), (3) only the candidates close to the query could form G', (4) the decision cost on each step.

Steiner-Hardness on All Graph Indexes

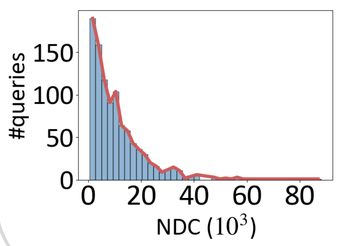
We use ME on MRNG as *Steiner*-hardness

- MRNG is pruned from KGraph
- Strict RNG pruning rule (no specific tricks)



Unbiased *Steiner*-Workload Generation

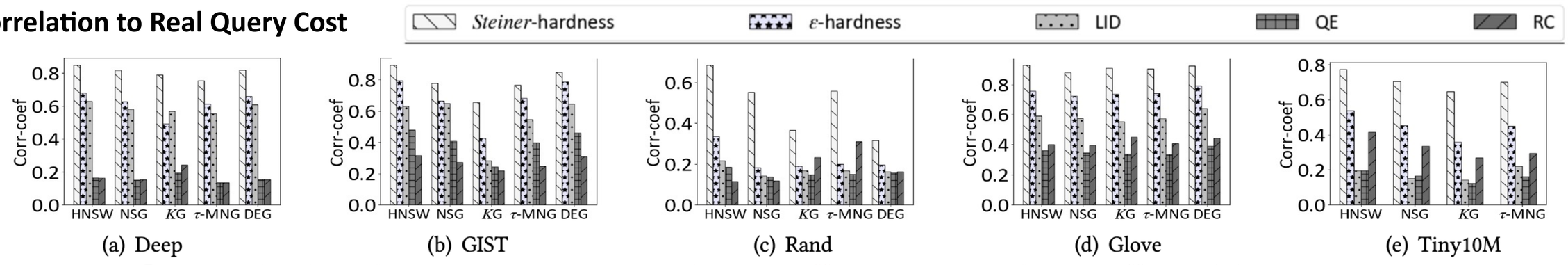
Current workloads are dominated by **easy queries**, leading to **over-optimistic** results.



1. Over-sample queries from the same data distribution
2. Calculate *Steiner*-hardness for new samples
3. Pick queries from different hardness ranges

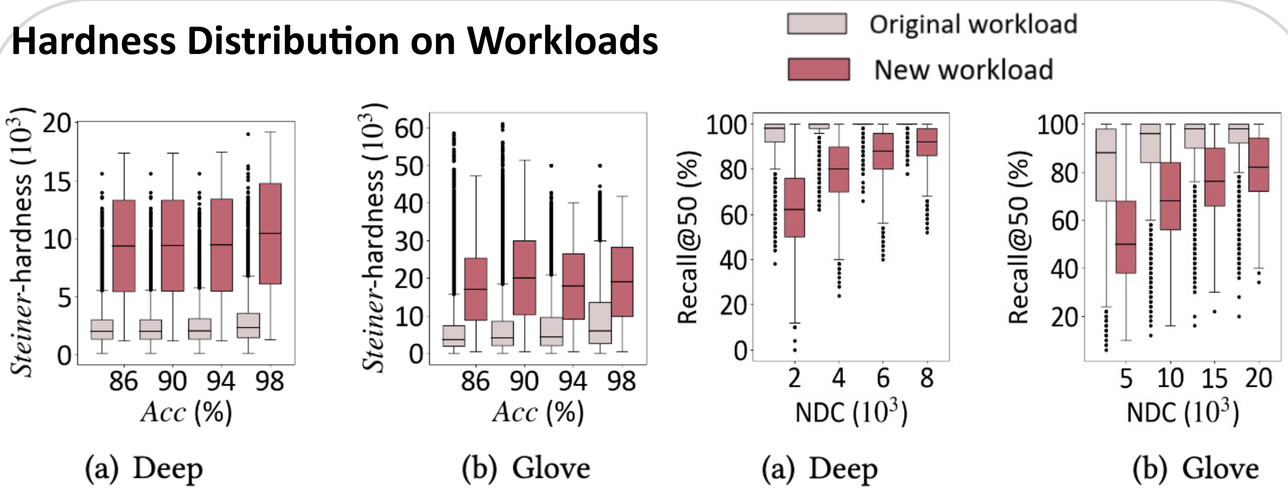
4 Experiments

Correlation to Real Query Cost



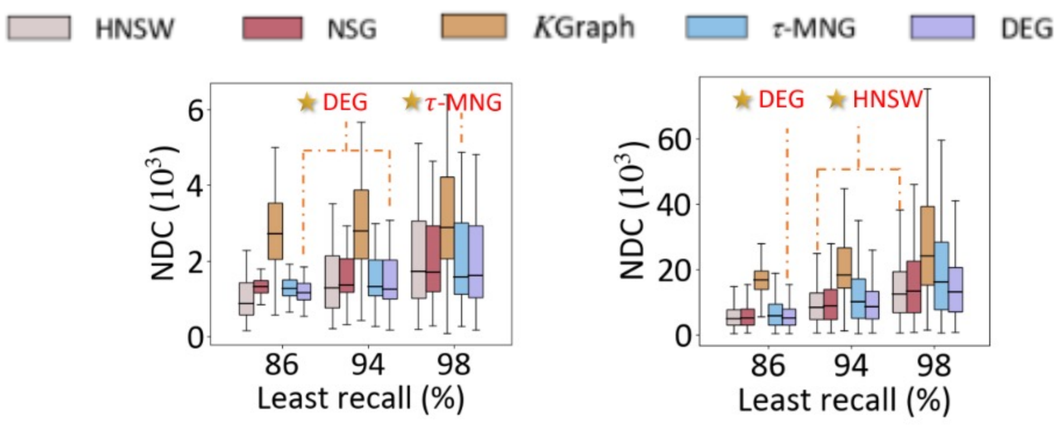
Steiner-hardness has on-average 0.75 correlation coefficient to the real query effort, while LID is 0.42.

Hardness Distribution on Workloads



*More hard queries are included in the **Steiner**-workload, making an even distribution of query hardness.*

Different Benchmark Results on Two Workloads (Deep1M)



On new workload: 10x worse avg. performance, larger performance variance, and $HNSW_0$ performs best.