Optimizing Fair Approximate Nearest Neighbor Searches using Threaded B+-Trees

Omid Jafari, Preeti Maurya, Khandker Mushfiqul Islam, Parth Nagarkar

New Mexico State University, Las Cruces, New Mexico

14th International Conference on Similarity Search and Applications, SISAP 2021

Sept 29 – Oct 01, 2021

INTRODUCTION

- Fairness can be divided into two categories:
  - Individual Fairness:
    - Treat individuals similarly
  - **Group Fairness:**
    - Treat groups of individuals similarly
- **Locality Sensitive Hashing** (LSH)-based techniques are very popular approximate Nearest Neighbor (NN) techniques for high-dimensional spaces

MOTIVATION

- LSH methods
  - provide theoretical guarantees
  - are data-independent
  - are scalable
- Existing LSH methods treat all dataset points similarly in the indexing and query phases

NAÏVE STRATEGIES

- **Assumption:** two groups of points in the dataset (A and B)
- **Strategy 1:** Divide dataset into two separate datasets
  - Not space efficient
  - Redundant processing
- **Strategy 2:** Change stopping conditions of LSH to continue searching until enough points are found from each group of points
  - Extra and unnecessary data is read from the indexes

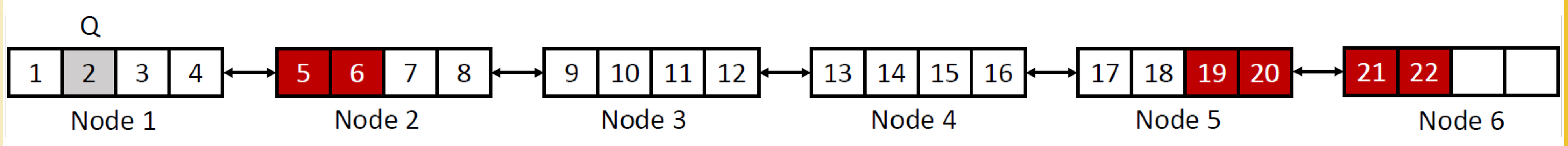
CONTRIBUTIONS

FairLSH (Our index structure for finding **Fair** approximate nearest neighbors using **L**ocality **S**ensitive **H**ashing)

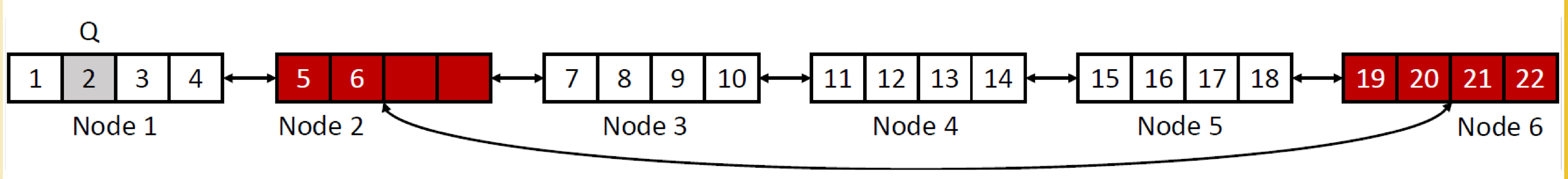
- FairLSH-Basic: Threaded tree structure to reduce disk I/O
- FairLSH-Advanced: Cost model-based tree structure to tune the trade-off between I/O costs and processing times

FairLSH-Basic

- **Intuition:** Unnecessary points should be skipped when reading indexes
- QALSH index structure:

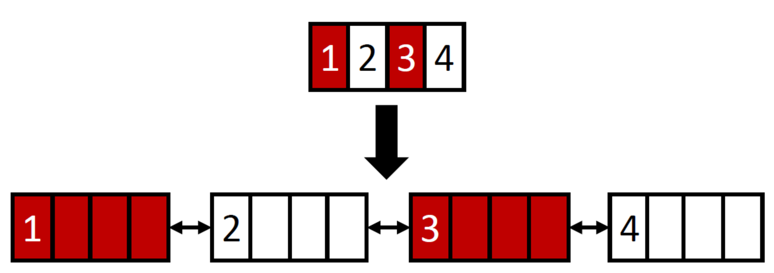


- **FairLSH-Basic index structure:**



Drawbacks

- Processing overhead of extra pointers no longer offsets I/O cost savings when points from different groups are sparsely distributed in the nodes



FairLSH-Advanced

- Detects how breaking down a node affects overall performance
- Costs related to reading nodes:
  - C_s: Cost of disk seeks
  - C_h: Cost of reading the header
  - C_p: Cost of reading the payload
- Total cost before breaking down a node:
  - $C_{Before} = C_{s,B} + C_{h,B} + C_{p,B}$
- Total cost after breaking down a node:
  - $C_{After} = C_{s,A} + C_{h,A} + C_{p,A}$
- Cost function:
  - $\begin{cases} break\ down, & if\ CAfter - CB_{efore} \leq \theta \\ do\ not\ break\ down, & if\ C_{After} - CB_{efore} > \theta \end{cases}$
  - θ is a user-defined parameter

EXPERIMENTAL SETUP

ALGORITHMS

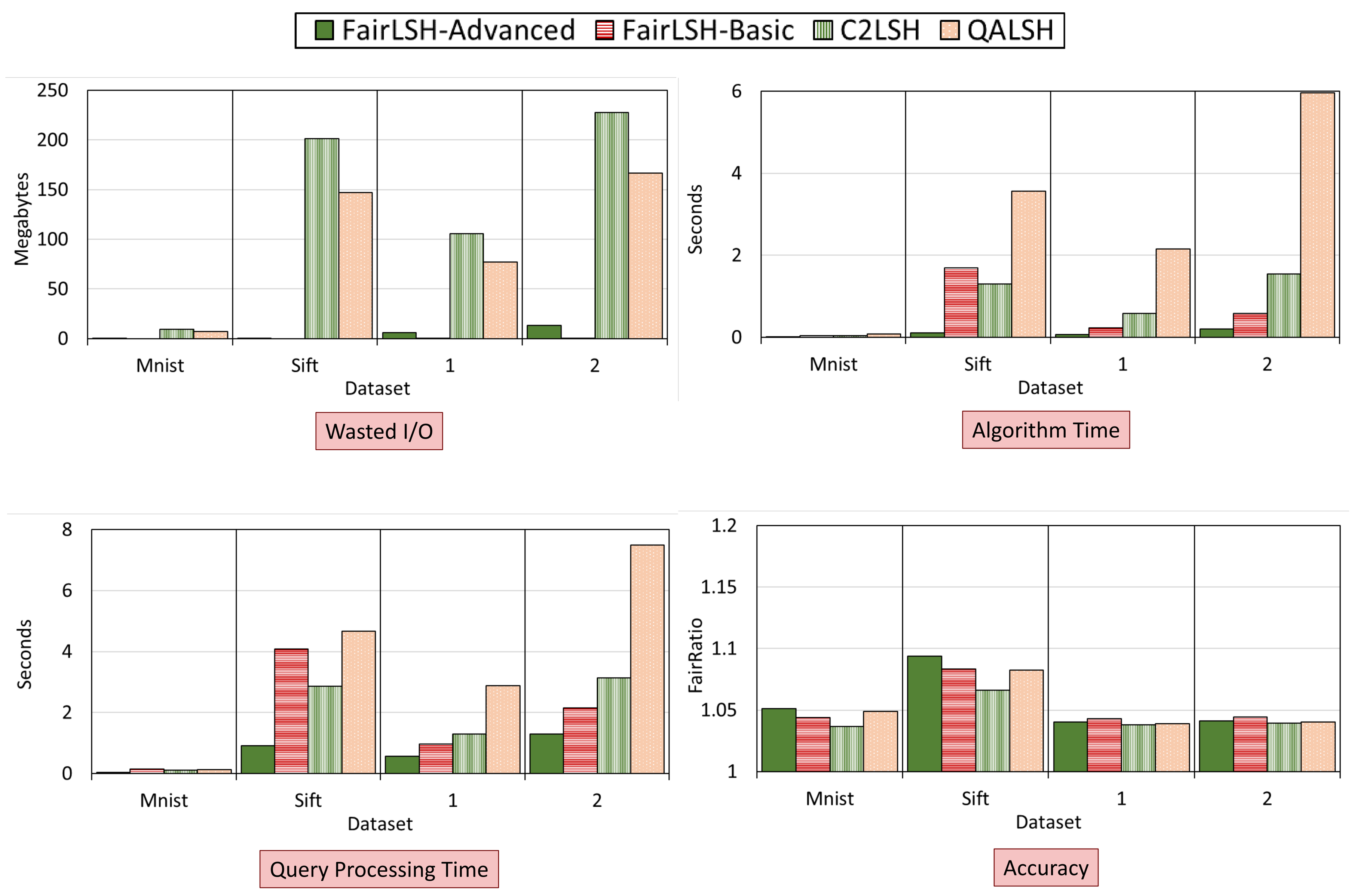
- C2LSH
- QALSH

CRITERIA

- Wasted I/O size
- Algorithm Time
- Query Time
- FairRatio

Dataset	Cardinality	Dims.
Mnist	60,000	50
Sift	1,000,000	128
Synthetic 1	500,000	1,000
Synthetic 2	1,000,000	1,000

RESULTS



CONCLUSION

- We presented FairLSH that can efficiently find fair top-k approximate nearest neighbors using LSH
- We presented two novel strategies that use threaded B+-trees and advanced cost models to improve the performance of LSH