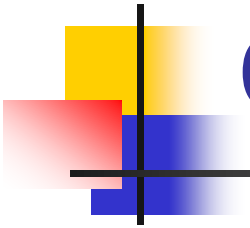
NEST: Locality-aware Approximate Query Service for Cloud Computing

Yu Hua, Bin Xiao[+], Xue Liu[*]

Huazhong University of Science and Technology

[+]The Hong Kong Polytechnic University

[*]McGill University

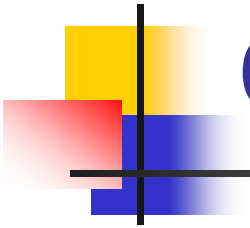


Outline

- Motivations and Backgrounds
- Design and Implementations
- Performance Evaluation
- Conclusions

Cloud Applications

- **Very large amounts:**
  - 1.8ZB in 2011, 40ZB in the next 8 years from IDC report
- **Heterogeneous types:**
  - Text, web, image, audio, video, etc.
- **Weaker locality:**
  - Various applications sharing a cache



Query is important

- Query is **the premise** of various operations in cloud applications
- Satisfy users' requests
- Locate a specified file
- Support "delete", "modify", etc.

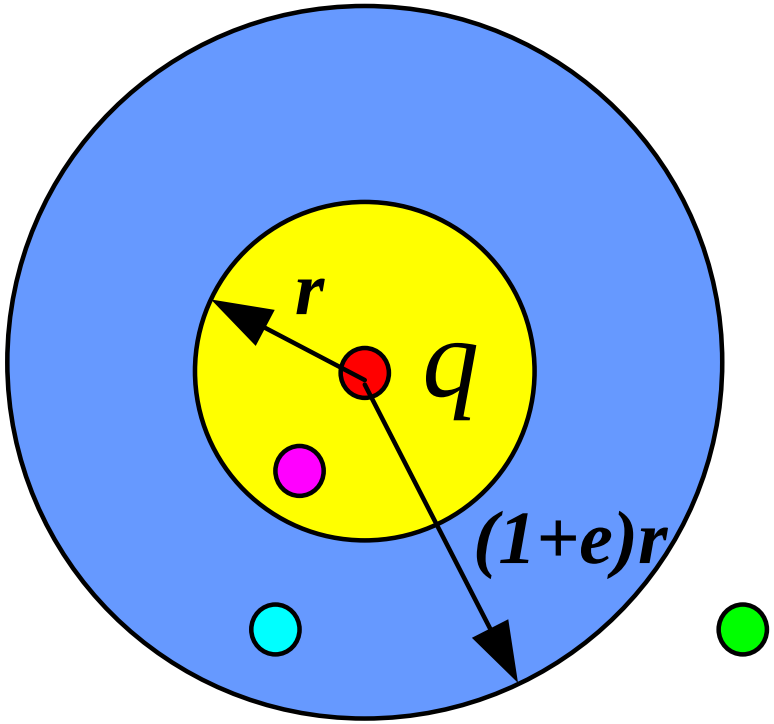
ANN Query

- Common query input is **not very clear** due to:
  - Large amounts of data
  - Inaccurate description
  - Typo
  - Multiple dimensions (time, directory, name, created time, modified time, etc.)
- ***Approximate Near Neighbor (ANN) Query is a suitable choice for cloud applications***

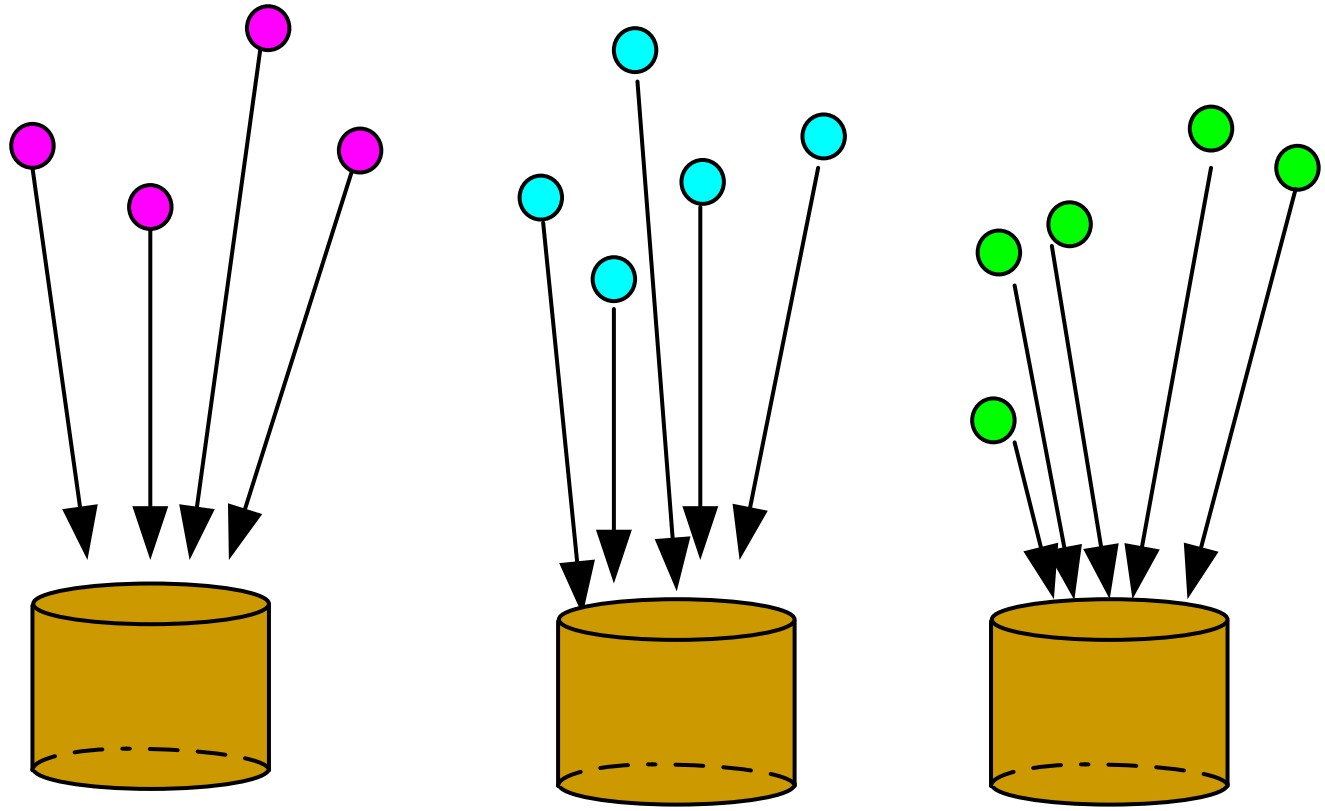
Locality Sensitive Hashing (LSH)

Near neighbor?

- ***yes***
- ***not sure***
- ***no***

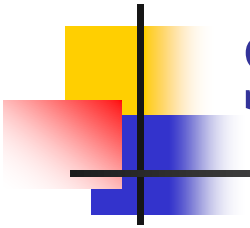


Locality Sensitive Hashing (LSH)



Locality Sensitive Hashing (LSH)

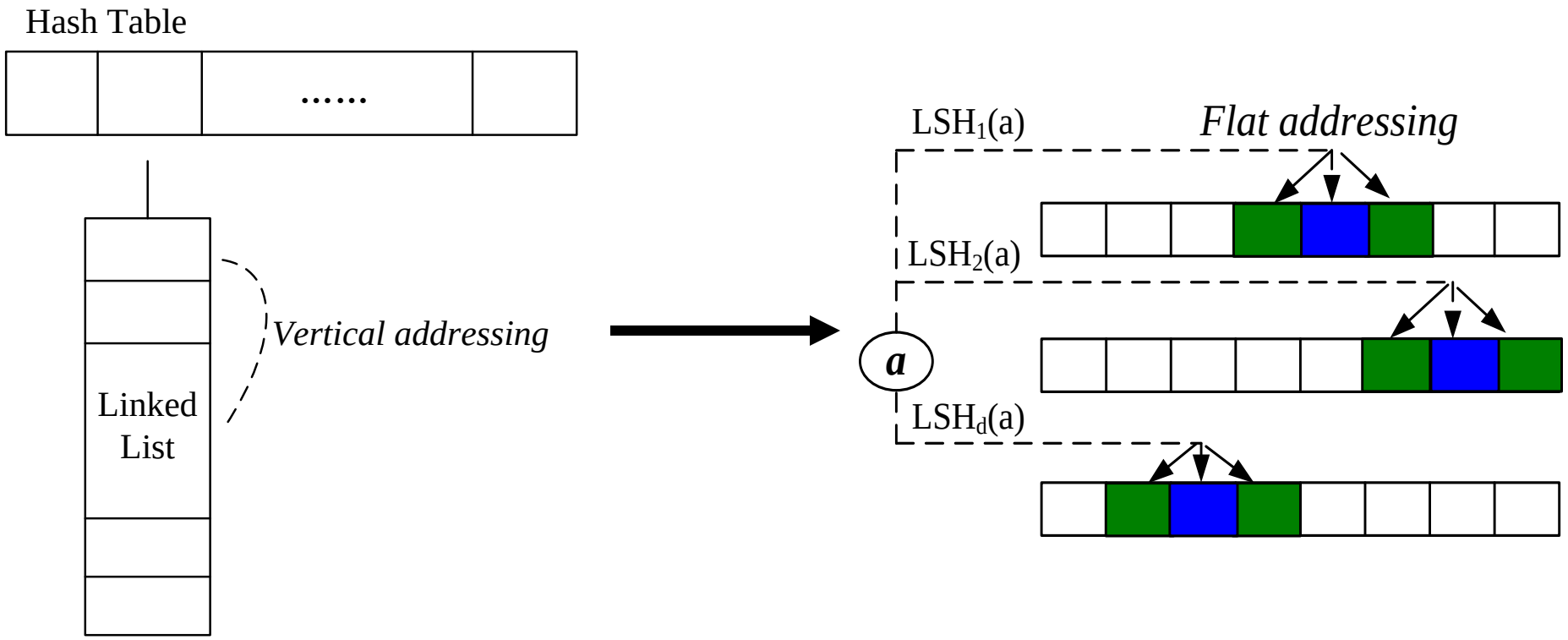
- **Benefits:**
  - Support ANN query
  - Simple hash computation
  - Faithful maintenance of data locality
- **Drawbacks:**
  - Space inefficiency
  - Imbalanced load in the buckets



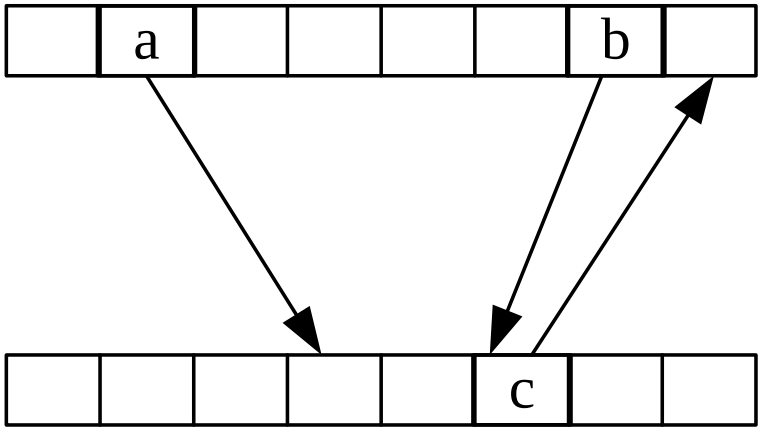
Solutions

- Address the **space inefficiency**:
  - *Transform vertical addressing into flat and manageable addressing*
- Address the **imbalanced load:**
  - *Cuckoo-driven approach*

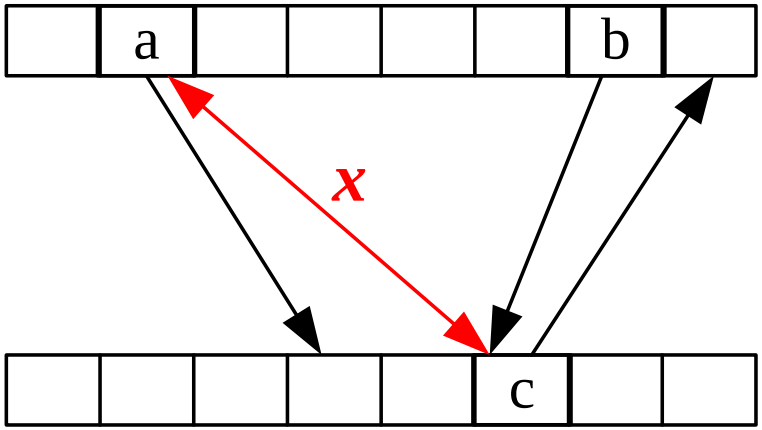
Vertical and Flat Addressing



Cuckoo-driven Way

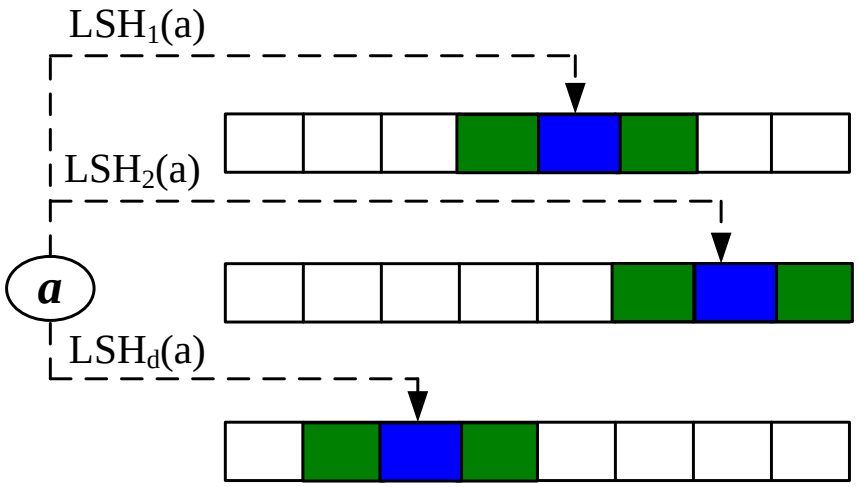


Initial status

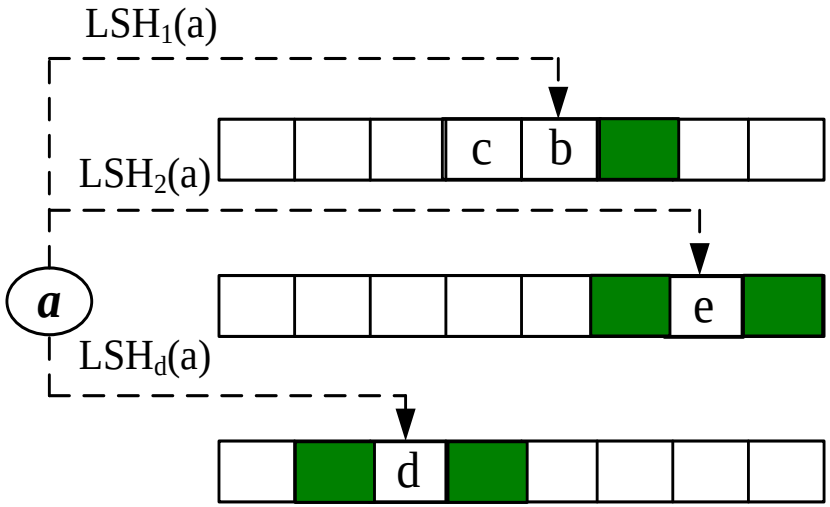


One hash collision

Cuckoo-driven LSH

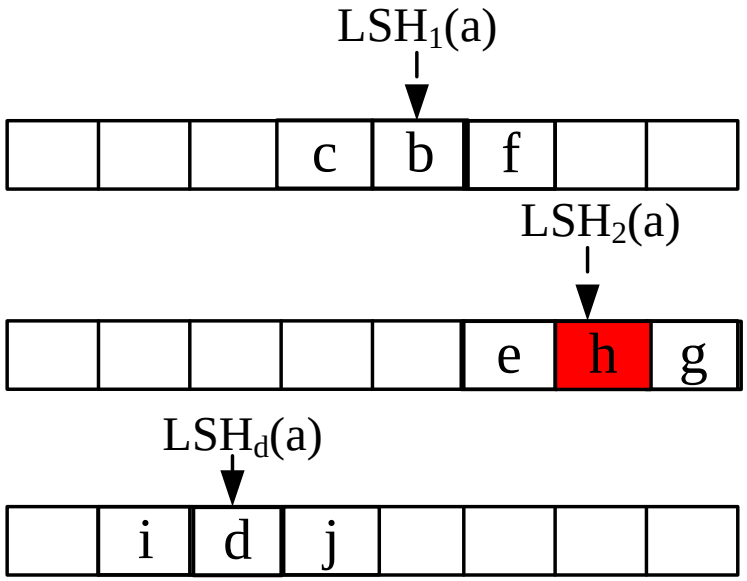


A multi-choice LSH

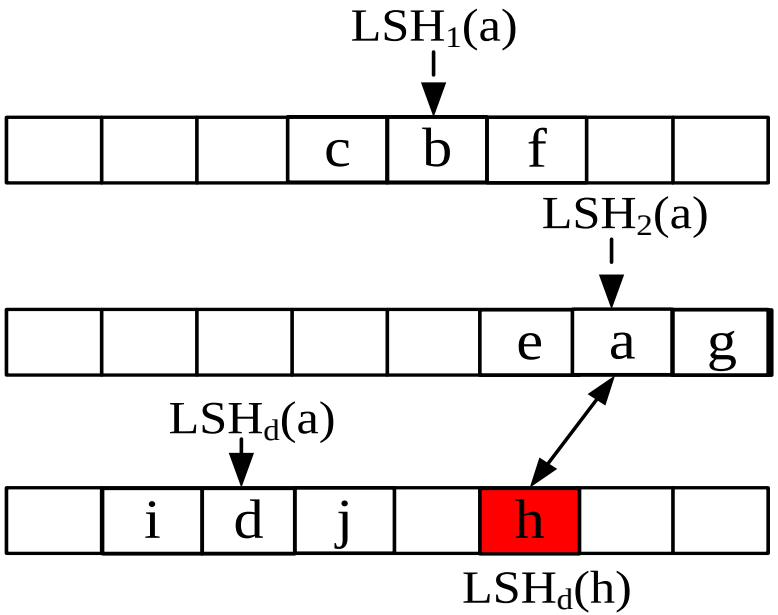


Available locations for item *a*

Cuckoo-driven LSH



Hashing collisions for placing item *a*



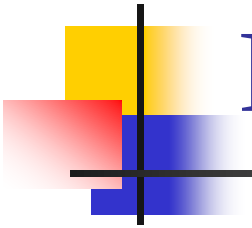
Moving item *h* to its another location

Support Practical Operations

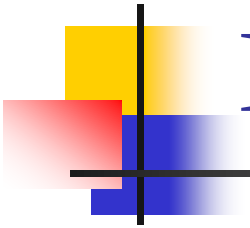
- Insertion
- ANN Query
- Deletion

See the details of algorithms and theoretical analysis in our paper

Performance Evaluation



- Real implementation in a large-scale cloud computing environment
- 100 servers (Intel 2.0GHz dual core CPU, 2GB DRAM, 250GB disk)
- Prototype is developed in the Linux kernel 2.4.21 environment



Evaluation Metrics

- Accuracy of ANN query
- Latency of ANN query,
- I/O costs
- Space overhead
- Rehash probability

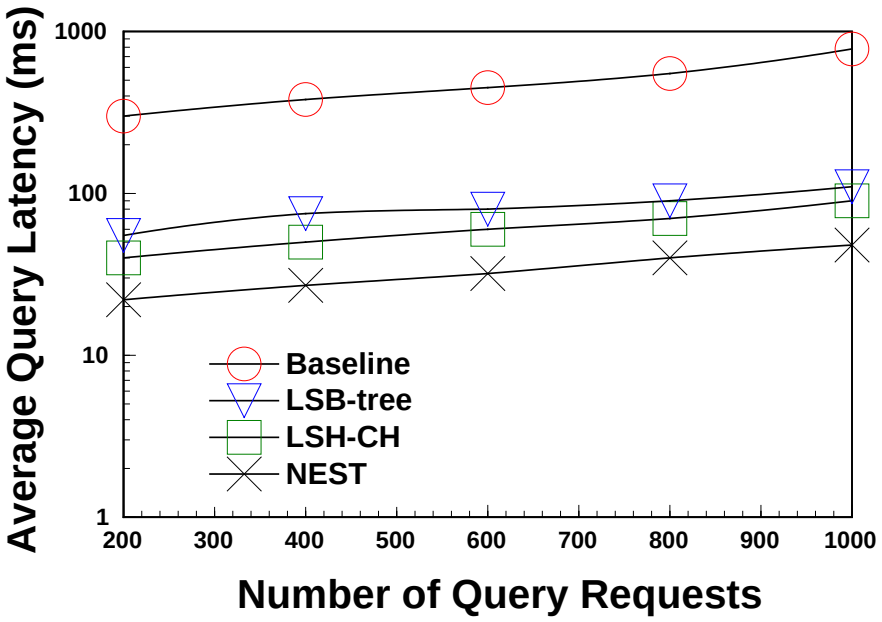
Real-world trace

- ***Microsoft* metadata trace:**
- From 2000 to 2004
- More than 63,398 distinct file systems
- 4 billion files (one of the largest sets of file system metadata collected)
- The 92GB-size trace has been published in SNIA

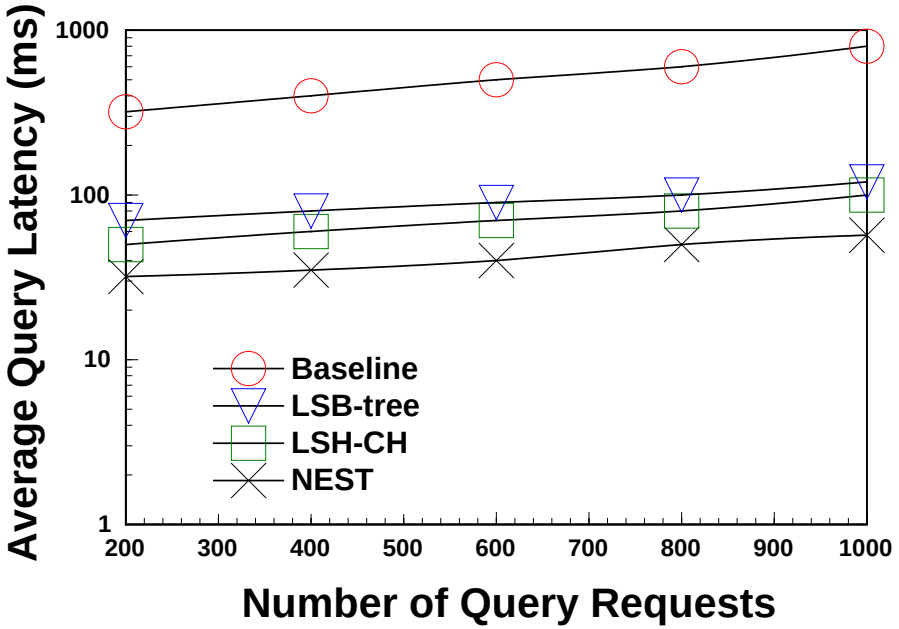
Query Requests and Comparisons

- **Requests:**
  - Generated from the attribute space of trace
  - Randomly selected by considering 1000 uniform or 1000 zipfian
  - Parameter *H* is 0.75
- **Comparisons:**
  - LSB-tree [SIGMOD 2009]
  - LSH with Cuckoo Hashing (LSH-CH): *simple combination*
  - Baseline approach

ANN Query Latency



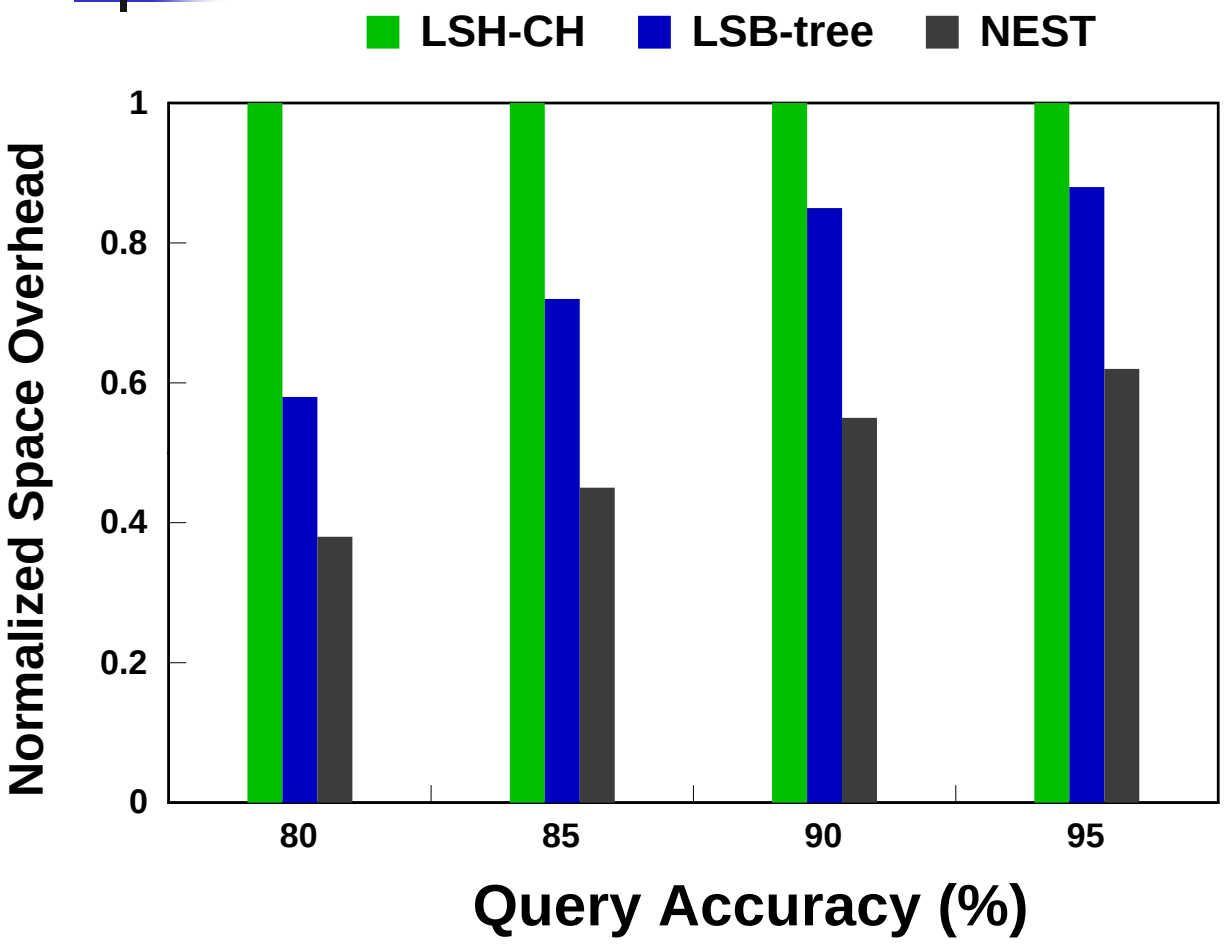
Uniform



Zipfian

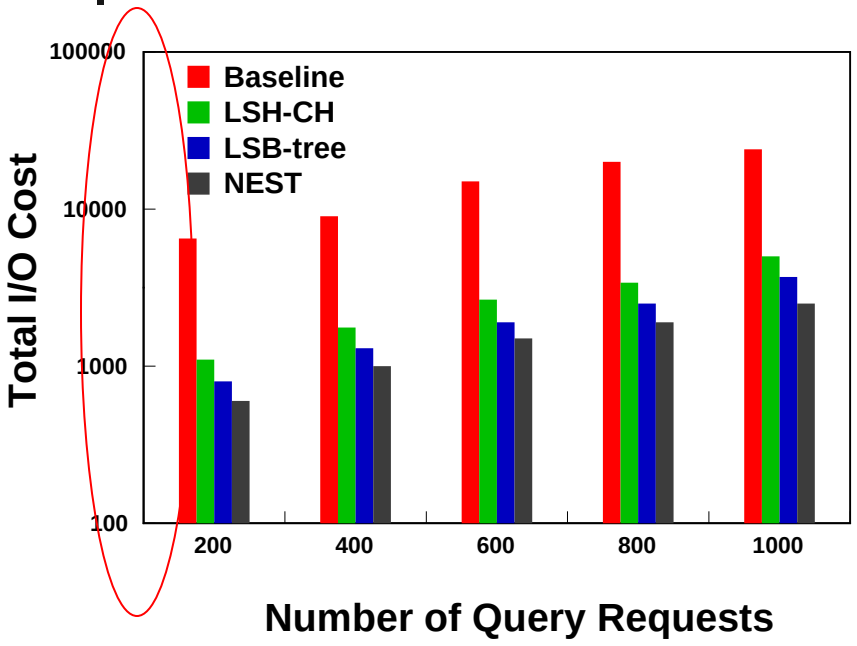
More than 40% time savings compared with state-of-the-art work

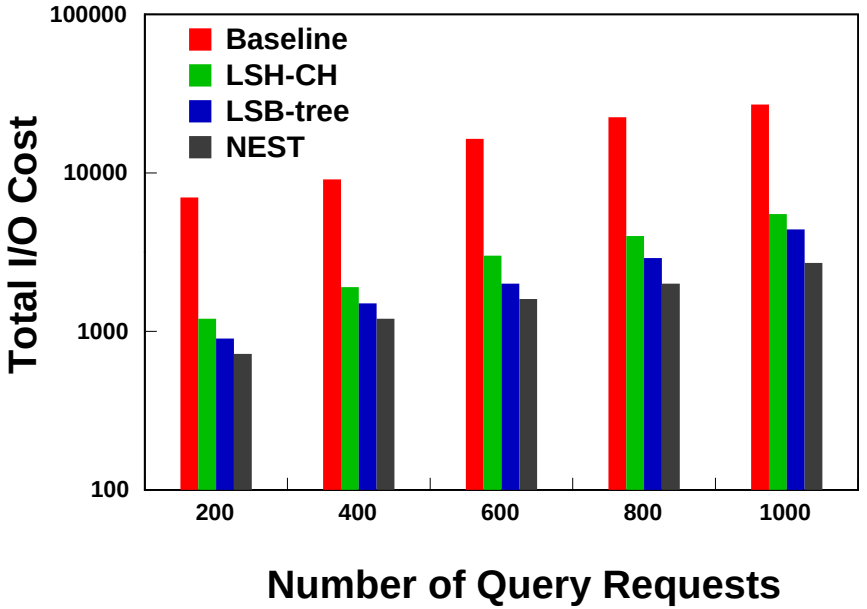
Space Overhead



Significant space savings allow us to put more indexes into high-speed memory

I/O Costs





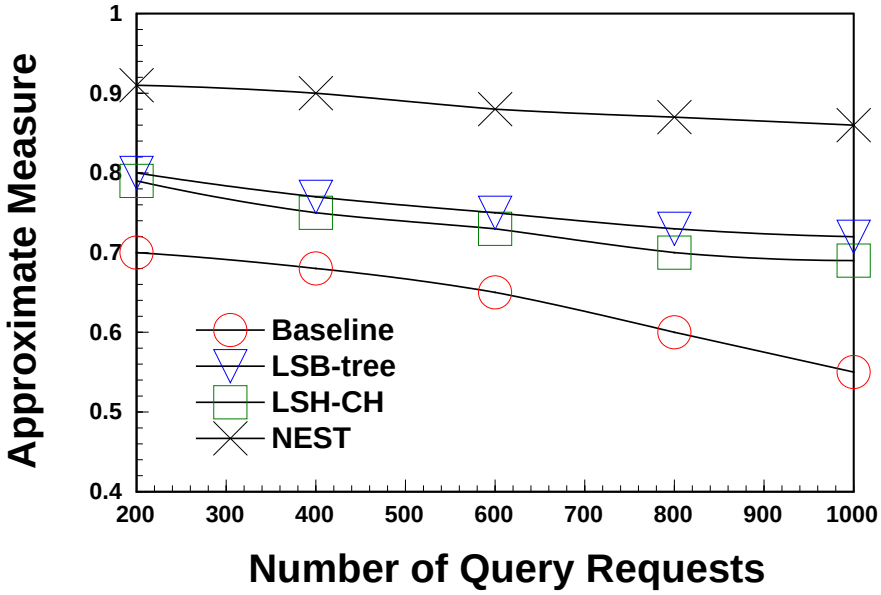
Log scale

Uniform

Zipfian

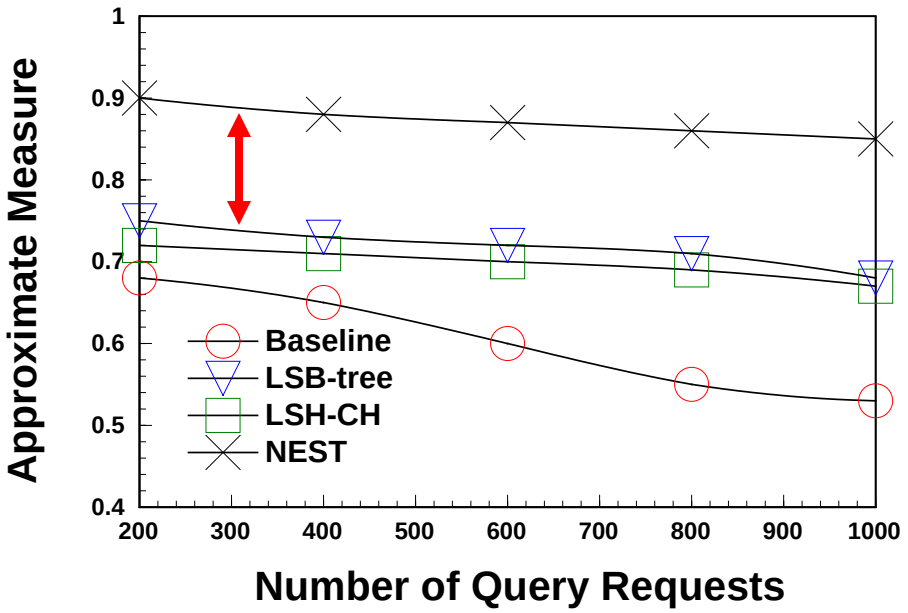
Low I/O costs from infrequent accesses to low-speed hard disks

ANN Query Accuracy



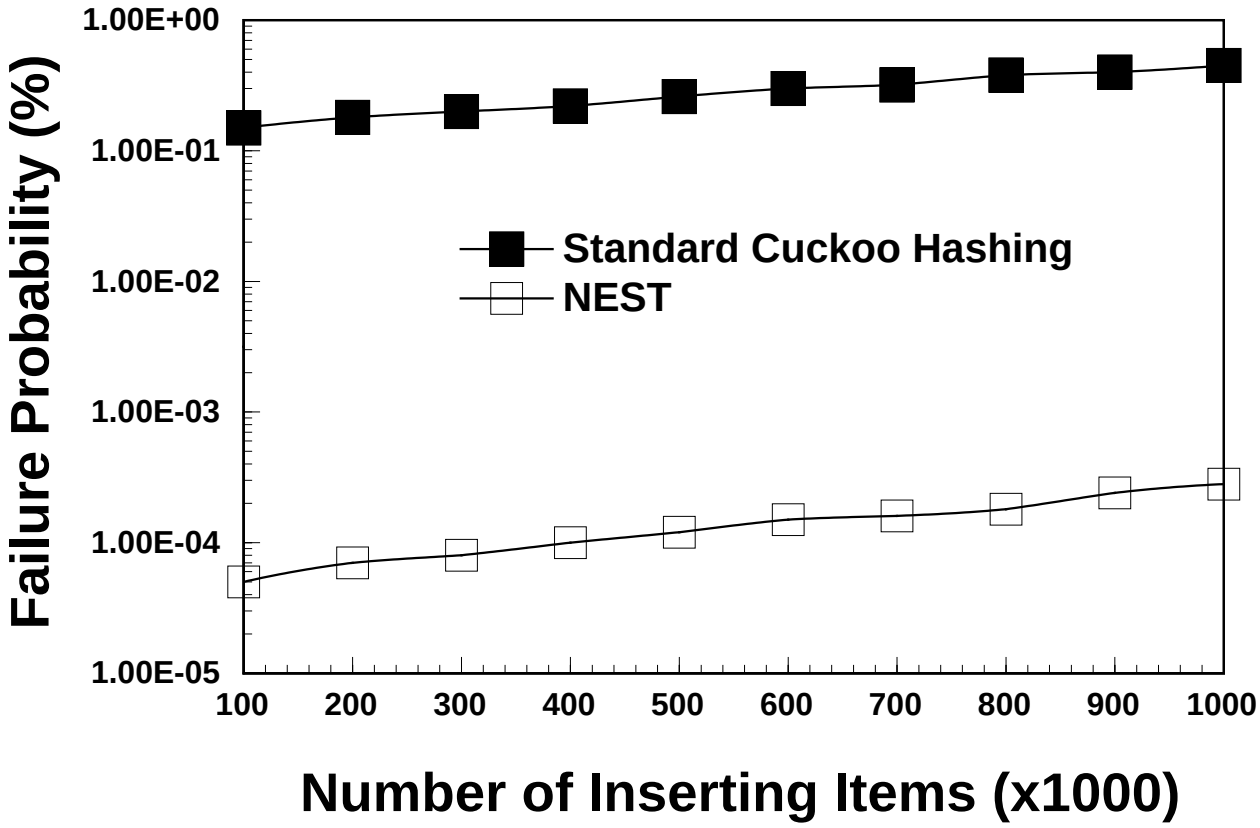
Uniform

Over 8% accuracy improvements



Zipfian

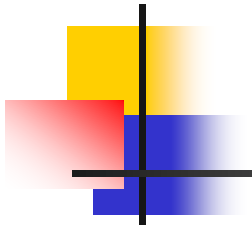
Rehash Probability



A failure only occurs from millions of insertions

Conclusion

- NEST is a locality-aware hashing scheme
- Address two important problems
- Obtain a fast and limited flat addressing with O(1) complexity
- Efficiently support ANN query service for large amounts of data



Thanks & Questions