XSEARCH: BENCHMARK THE STATE-OF-THE-ART DISTRIBUTED SEARCH PLATFORMS (ON DISK)

- **CS-554: Data-Intensive Computing**
- **Team Members**
  - Nilesh Jorwar (A20405042)
  - Ashish Ryot (A20405230)
- **Project Mentor**
  - Alexandru Iulian Orhean

PROBLEM STATEMENT

- Benchmark information retrieval libraries Lucene and LucenePlusPlus on single node and multi-node system.
- Improvisation of Indexing and Searching using Parameter tuning on Lucene and LucenePlusPlus
- Multi-node cluster setup for Solr Cloud with Zookeeper

MOTIVATION

- Continuous growth in scientific and unstructured data (in exponential terms) necessitates demand for faster information retrieval.
- Available search engines perform better on distributed file systems and well-structured data, but fails to provide timely response and support for complex and lengthy search queries.

RELATED WORK

- Various studies and researches are going on the in the field of distributed search platforms indexing improvement. One of search engines available, a slow search, primarily deals with high quality search experience while relaxing speed requirements [1]. Although this search engine gives us new search experience with high quality result but does not focus on the time factor considering the vast amount of data to search for indexing.
- There is another paper which provides an alternative solution, the use of dynamic indexing in search engine, where indexing process is distributed over the cluster of computers in grid computing to improve the performance through distributed load [2]. Although the process of distributing the indexed data over distributed memory helps faster indexing for small amount of data, say few petabytes of files, but significantly reduces the chances of faster data retrieval considering the millions of data resided on distributed storage systems.

GOALS

- To benchmark Lucene and LucenePlusPlus
  - To achieve higher indexing throughput
  - To reduce search time
  - To lower the index size
- To compare the performance of Lucene and LucenePlusPlus with Apache Solr and SolrCloud

PROPOSED WORK: ARCHITECTURE

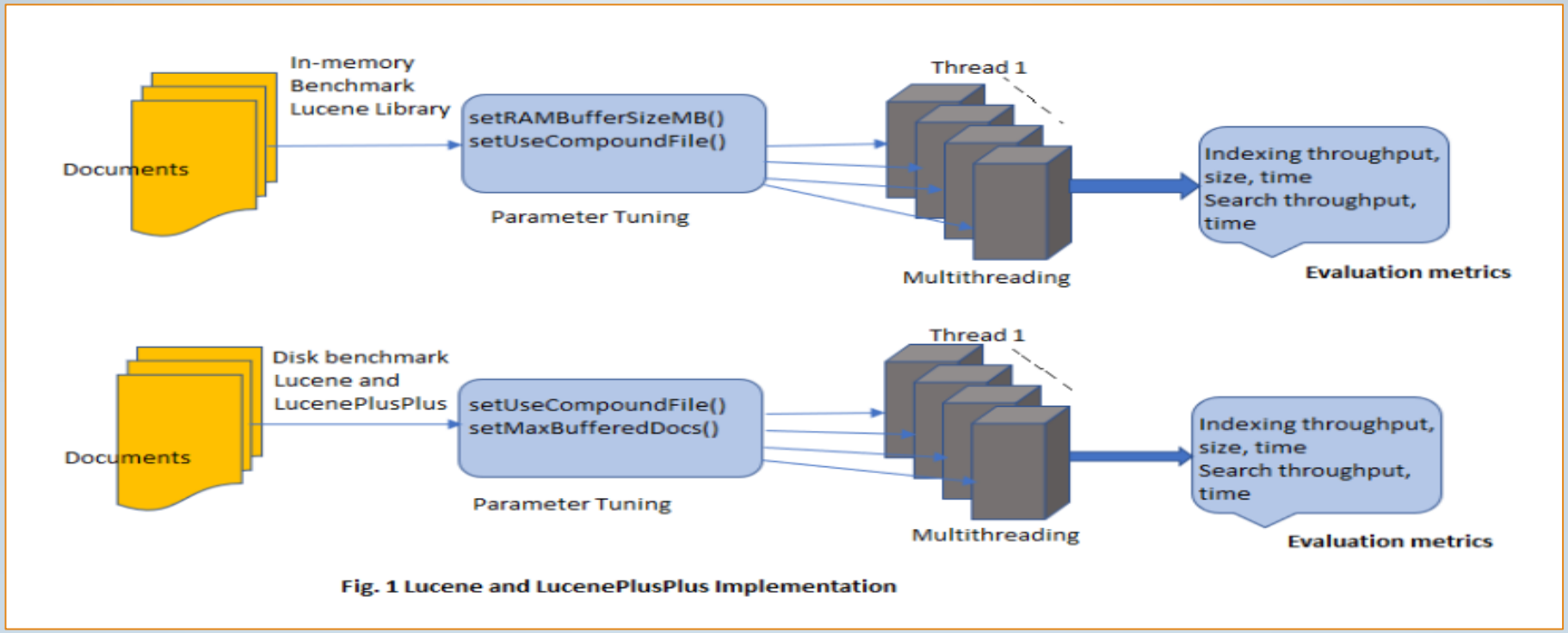


Fig. 1 Lucene and LucenePlusPlus Implementation

Lucene/LucenePlusPlus

- Achieved….
  - Ability to index up to 200GB metadata dumps on Chameleon Cloud’s Bare Metal Instance
  - Multithreading
  - Parameter tuning
  - Smaller index size and higher throughput
  - Higher query throughput
- Challenges
  - Opening > 65K files resulted in runtime errors.
  - Setting up multi-node cluster for SolrCloud Zookeeper

PROPOSED WORK: SolrCloud With Zookeeper

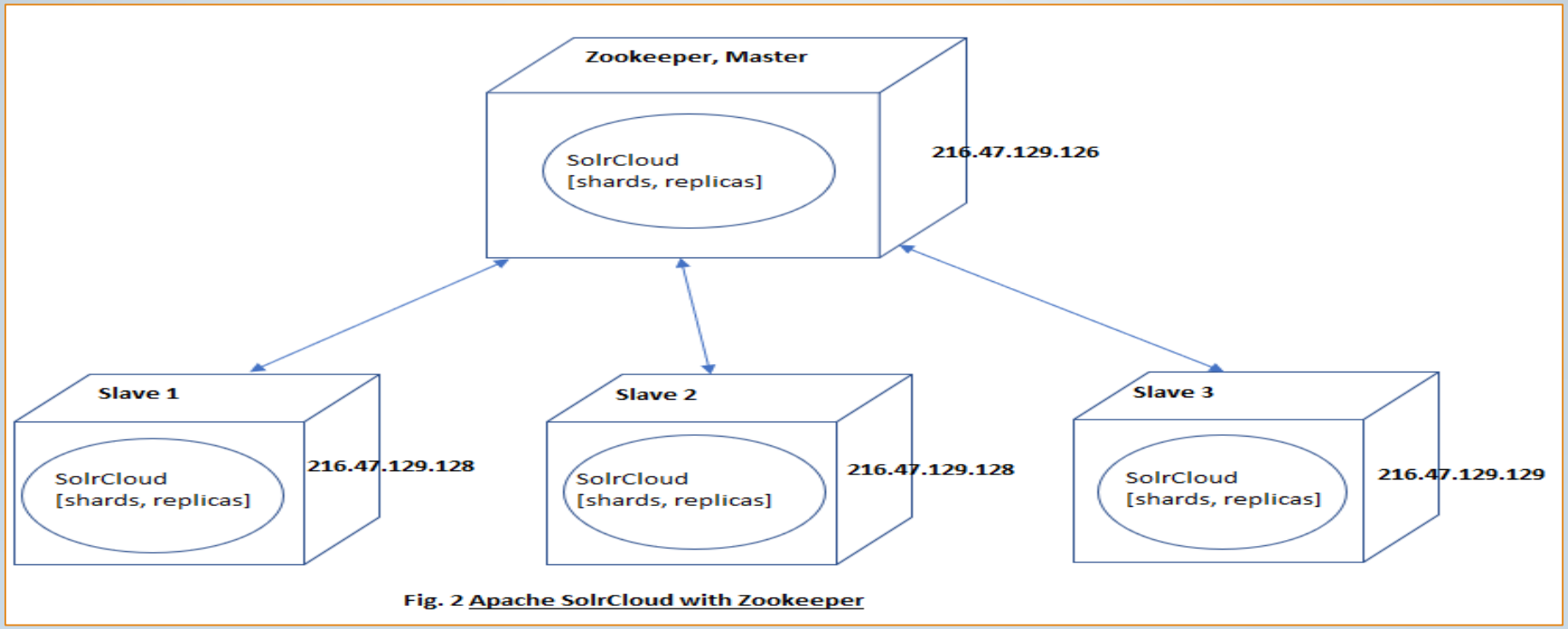


Fig. 2 Apache SolrCloud with Zookeeper

SolrCloud With Zookeeper

- Configuration
  - Download Latest zookeeper-3.4.10 on Master and all slaves and unzip it
  - Copy conf/zoo_sample.cfg to conf/zoo.cfg and change it to include following parameters
    - clientPort=2181 (Solr uses this port to access Zookeeper)
    - initLimit=10
    - dataDir=/var/lib/zookeeper
    - autopurge.snapRetainCount=4
    - autopurge.purgeInterval=24
    - syncLimit=2
    - server.1=216.47.129.126:2888:3888
    - server.2=216.47.126.126:2888:3888
    - server.3=216.47.137.102:2888:3888

Continued…

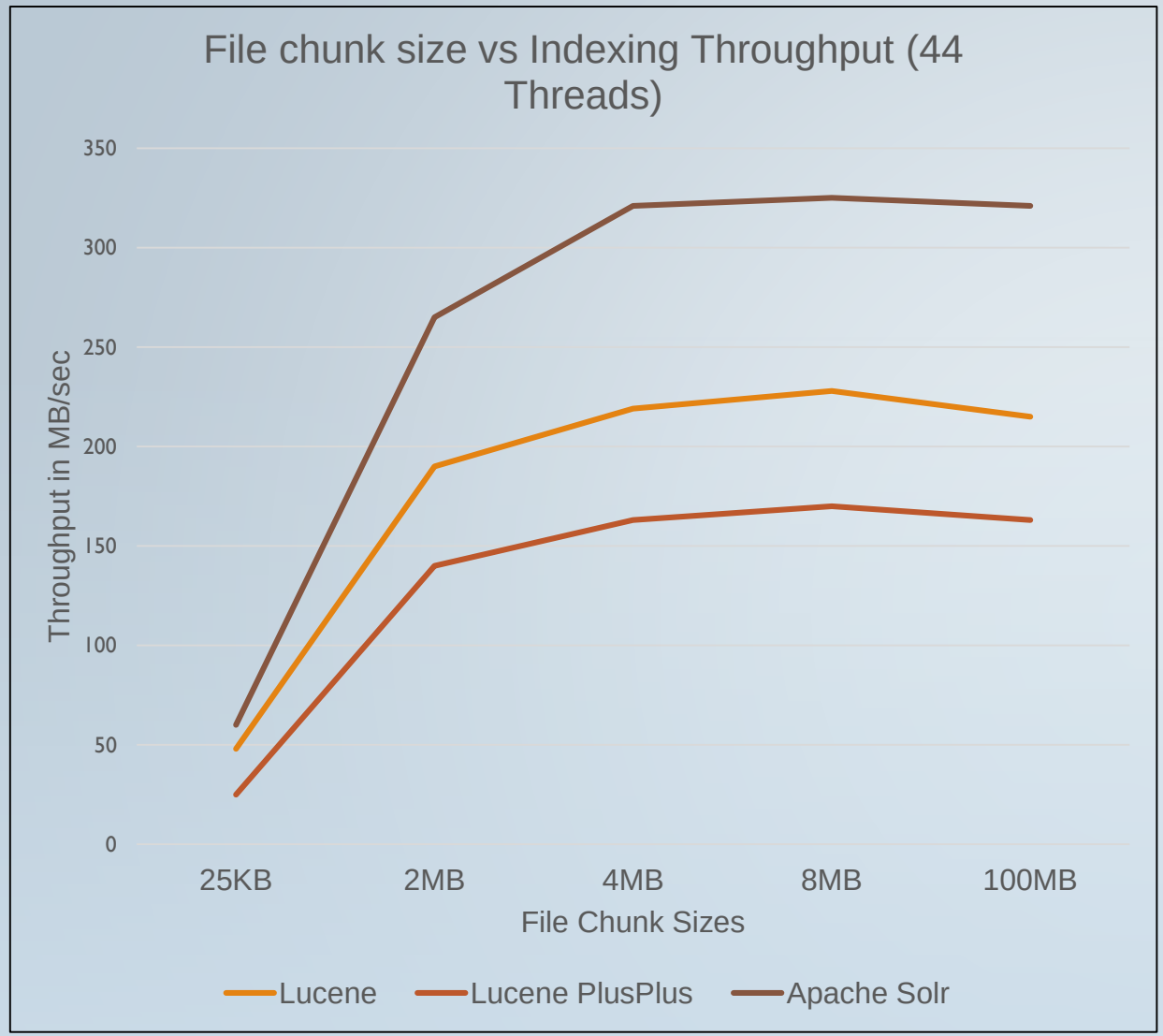
- Create directories /var/lib/zookeeper on each slave and create myid file and write 1,2 and 3 respectively in myid file of each slave.
- Start the zookeeper on each slave using following command from /bin folder
  - ./zkServer.sh start-foreground conf/zoo.cfg
- Check the status of these slaves (zookeeper) using ..
  - echo status | nc ip_address 2181 …(ip_address: =216.47.129.126 or 216.47.129.129 Or 216.47.134.102)
- Download latest Apache Solr in Cloud mode on separate machine and point it to the Zookeeper instances
- Change the solr.in.sh file to include the slave instances along with ports under ZK_HOST parameter or run the following command to start the Solr in Cloud mode
  - bin/solr start -cloud -s <path to solr home for new node> -p 8987 -z zookeeper_ip:zookeeper_port
  - bin/solr start -cloud -s “~/solr”-p 8987 -z “216.47.129.126:2181, 216.47.129.129:2181, 216.47.134.102:2181”
- Once solr is started and run the following command to report indexing time / sec.
  - java –DC=techproducts –jar post.jar dumps.dat

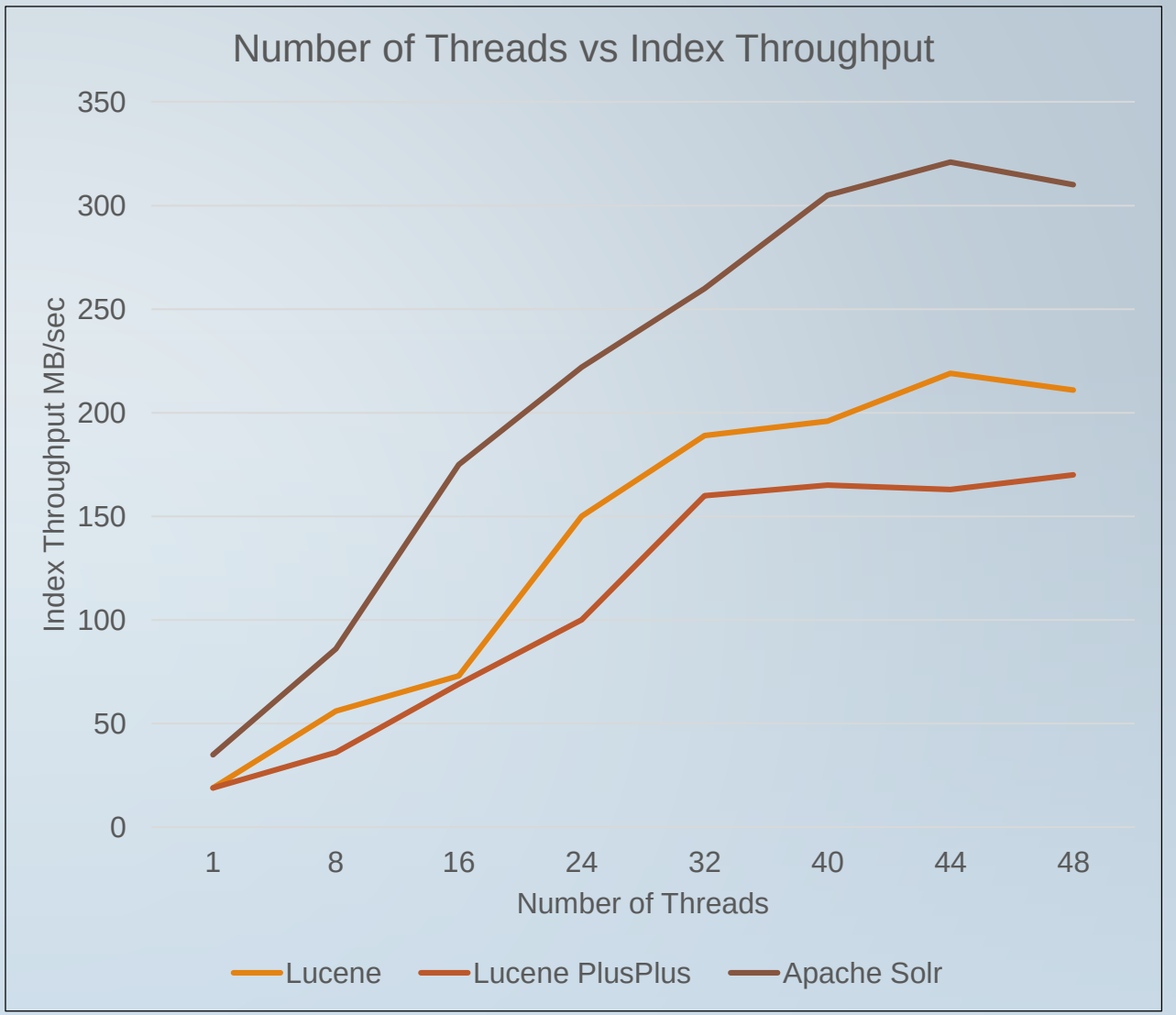
Continued…

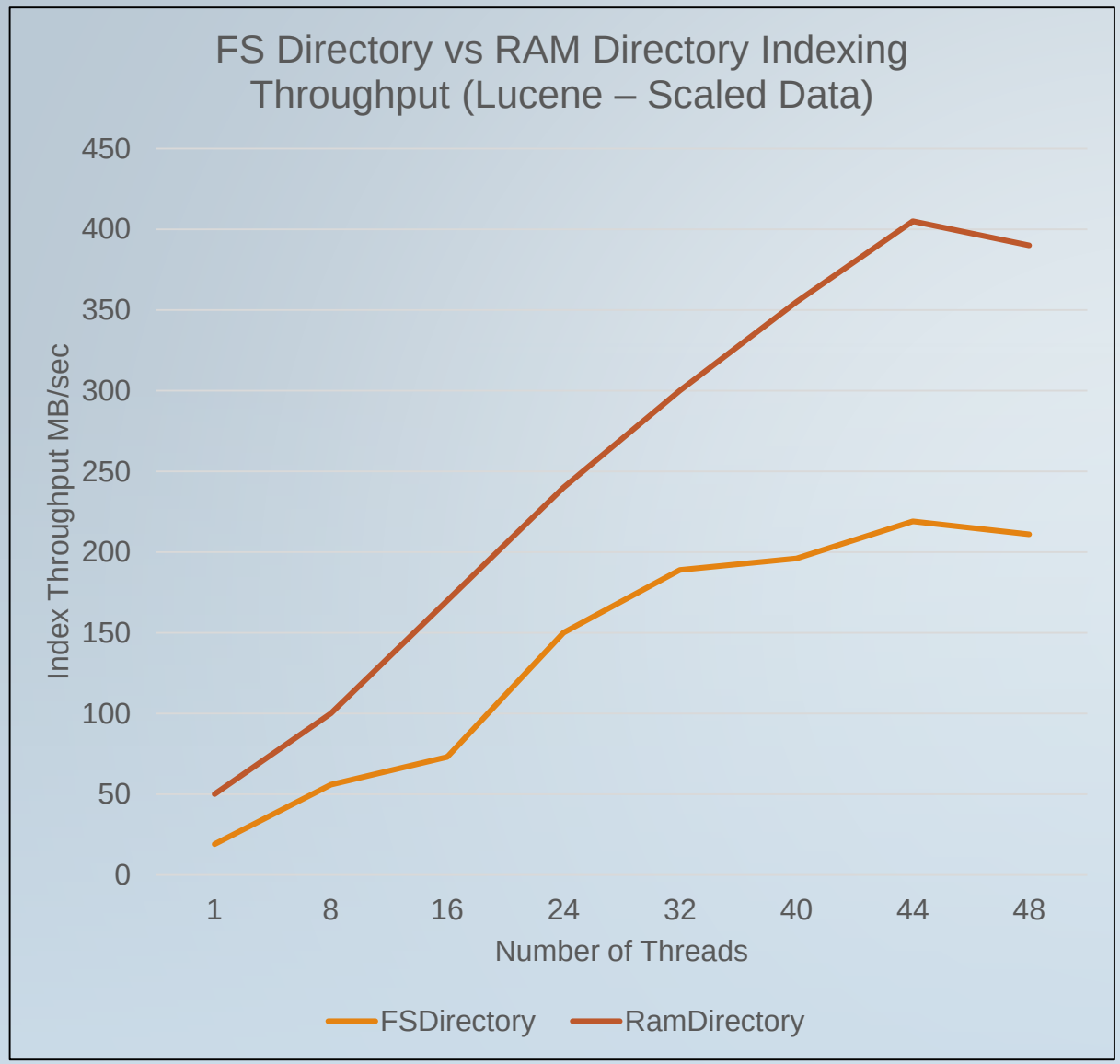
- Achieved …
  - Multi-node Apache Solr Benchmarking with Zookeeper
- Challenges
  - Multi-node cluster setup

PERFORMANCE EVALUATION

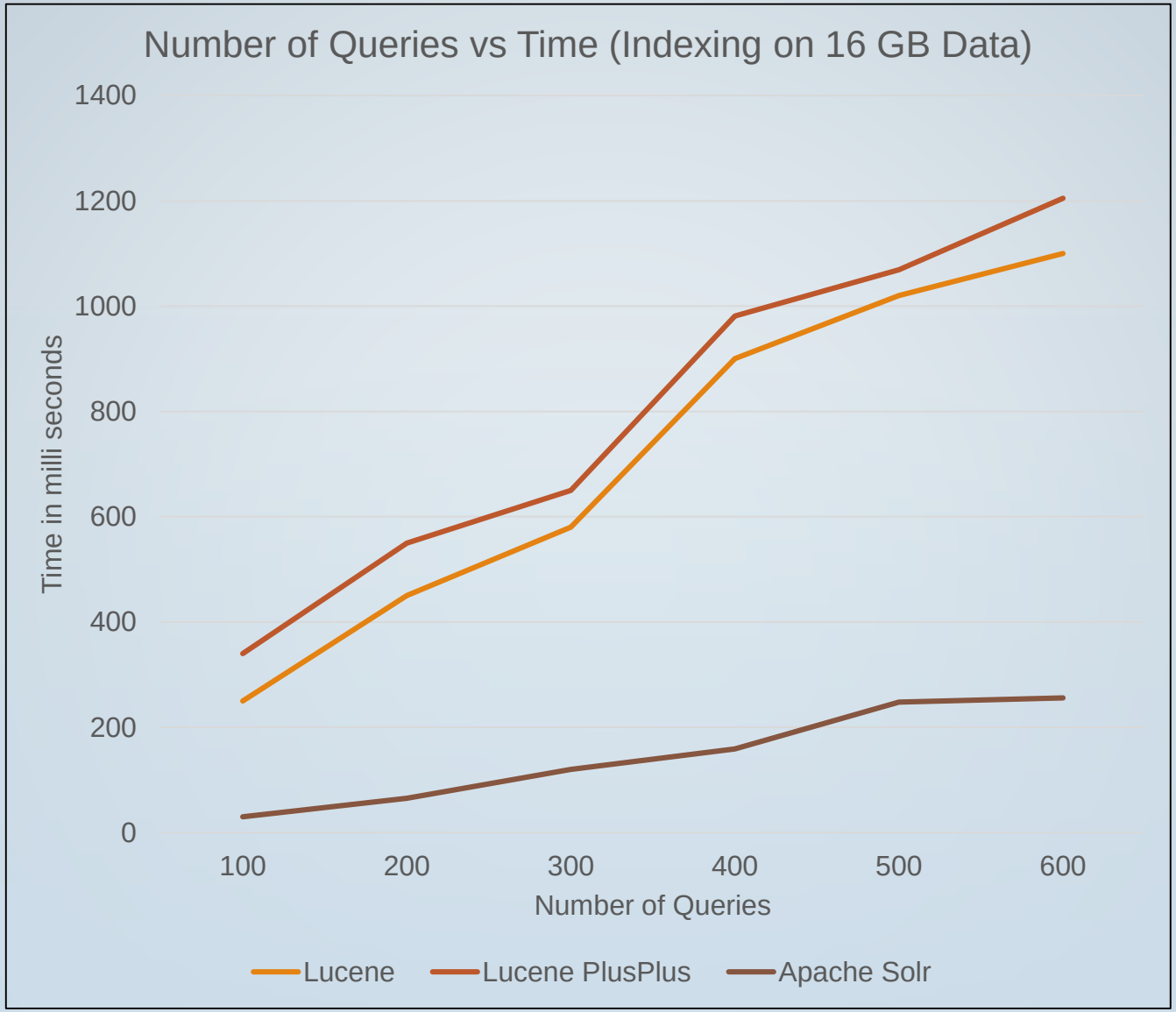
- Apache Lucene/ LucenePlusPlus
  - Benchmarked Lucene and LucenePlusPlus over the Chameleon Cloud's bare metal nodes.
    - Conducted experiments on scientific datasets of large size (~180GB)
    - Experiments on varying file sizes of 25KB, 2MB, 4MB, 8MB, 100MB, etc
    - Comparison between indexing on disk and memory
    - Reduced index size and increase in throughput
  - Comparison of Lucene and LucenePlusPlus with Apache Solr search engine.
- Solr with Zookeeper
  - Indexing Throughput- 20000 documents / sec
  - Low speed due to small sized files. Reading 1KB blocks from disk resulted in high latency.


File chunk size vs Indexing Throughput (44 Threads)
Throughput in MB/sec
350
300
250
200
150
100
50
0
25KB
2MB
4MB
8MB
100MB
File Chunk Sizes
Lucene
Lucene PlusPlus
Apache Solr


Number of Threads vs Index Throughput
Index Throughput MB/sec
350
300
250
200
150
100
50
0
1
8
16
24
32
40
44
48
Number of Threads
Lucene
Lucene PlusPlus
Apache Solr


FS Directory vs RAM Directory Indexing Throughput (Lucene – Scaled Data)
Index Throughput MB/sec
450
400
350
300
250
200
150
100
50
0
1
8
16
24
32
40
44
48
Number of Threads
FSDirectory
RamDirectory


Ratio of Data and Index size for Metadata dumps (44 Threads)
Ratio: Data Size/Index Size
4
3.5
3
2.5
2
1.5
1
0.5
0
2
4
8
16
32
64
128
180
Data Size in GB
Lucene
Lucene PlusPlus


Number of Queries vs Time (Indexing on 16 GB Data)
1400
1200
1000
800
600
400
200
0
Time in milli seconds
100
200
300
400
500
600
Number of Queries
Lucene
Lucene PlusPlus
Apache Solr

REFERENCES

- [1] Jaime Teevan, Kevyn Collins-Thompson, Ryen W. White, Susan T. Dumais & Yubin Kim, “Slow Search: Information Retrieval without Time Constraints” in Proceedings of HCIR 2013.
- [2] M. E. Elaraby, Omaima Nomir, M. Sakre, “Dynamic and Distributed Indexing Architecture in Search Engine using Grid Computing” in International Journal of Computer Applications (0975 – 8887), Volume 55– No.5, October 2012.