Finding a Needle in a Field of Haystacks

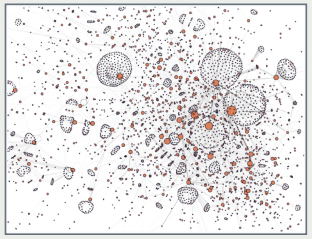
Lightweight Metadata Search for Large-Scale Distributed Research Repositories

Anna Blue Keleher, University of Maryland
Kyle Chard (Advisor), University of Chicago
Ioan Raicu (Advisor), Illinois Institute of Technology
Ian Foster (Advisor), University of Chicago
Alex Orhean (Advisor), Illinois Institute of Technology

1 ABSTRACT

- Fast, scalable, and distributed search services are increasingly available to organizations with large volumes of data to index, but often require that data be located centrally
- Globus, a distributed network for fast file transfers, lacks an efficient means for search
- Our solution: a two-level tree of lightweight metadata indexes that retains summary terms able to reference data while drastically reducing the amount of data stored centrally by upwards of 96% in all tests — a "second-level index" (SLI)

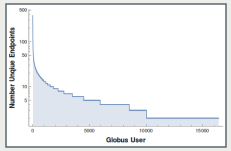
2 GLOBUS



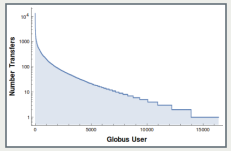
Globus file transfer network [1]. Nodes represent endpoints, scaled by data size, edges represent transfers (length is inversely proportional to transfer volume).

Globus by the Numbers

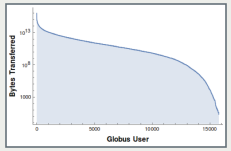
- 256 PB transferred
- 36 billion files processed
- 1 PB — largest single transfer to date
- 10,000 active endpoints
- 56,000 registered users
- 10,000 active users/year



Unique endpoints per Globus user



Number of transfers per Globus user



Bytes transferred per Globus user

Why metadata?

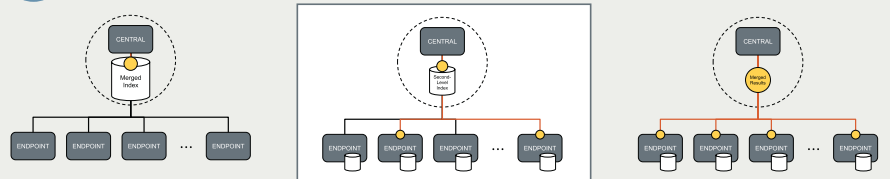
- Globus connects research repositories, many of which contain files that are not full text or easily indexable
- In such repositories, files are often characterized by long names and directory paths rather than contents
- Metadata is much easier to work with at scale — and Globus is huge

Which data?

- 37 largest public Globus endpoints (2.7 TB total)
- NERSC filesystem dump, similar to large private endpoints (164.6 TB total)

(Globus network images courtesy of William Agnew, Georgia Institute of Technology.)

3 DESIGN/SETUP



Traditional 1: single central index

DESCRIPTION
- Central index is result of merging indexes from each endpoint

ADVANTAGES
+ Users only need to store indexes when the endpoint's file system is updated

PROBLEMS
- Central index must store references to documents on endpoints (grows too fast)
- Heavy central storage burden

Our Method: hierarchical hybrid featuring "collapsed" second-level index (SLI)

- SLI references endpoints, and contains a summary subset of indexed terms
+ Some storage burden on users, but still very low per user
+ Lower storage burden on central servers
+ SLI returns a smaller subset of endpoints to which queries must be distributed

Traditional 2: fully distributed index

DESCRIPTION
- Indexes kept on each endpoint
- Central servers distribute query to each endpoint, then retrieve/merge results

ADVANTAGES
+ More space taken up by indexes than Method 1, but burden is split among users

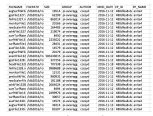
PROBLEMS
- Most search terms are likely to correspond only to a small subset of endpoints

Indexing with Xapian

- Xapian is an open-source indexing engine for full-text search with strong Python support
- We added new tokenization capabilities to allow it to interpret terms from filepaths, camel-case filenames, etc.
  `"IAmManyTerms.csv" -> F:'I',Am',Many',Terms'; E: 'csv'`
- Numerical metadata (sizes, datetimes) were indexed for range searches

NERSC FS Data Prep

- File system dump from February 2017
- Crawled FS and extracted metadata
- Divided metadata by "project," then binned into 8 equal-sized shards
- Distributed across 8 VM "endpoints"
- Binning by project prevents "endpoints" from being too similar



Sample metadata

4 RESULTS (PUBLIC ENDPOINTS)

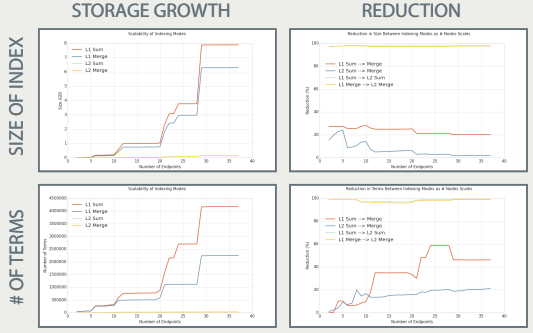
SLI Performance

	Level 1 Terms	Level 2 Terms	Reduction
Sum	11,602,821	59,983	99.48%
Merge	8,244,805	5,155	99.94%
Reduction	28.94%	91.41%	

SLI Performance (above)
- "Sum": sum of terms in all individual indexes
- "Merge": terms in the single merged index (built from individual indexes)

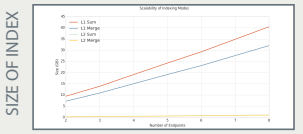
SLI Scalability (right)
- x-axis: number of endpoints (in arbitrary order) used to build the merged index
- Note: in "Storage Growth" graphs, the yellow line showing scalability of second-level index (sum) exactly overlaps green line (merge)

SLI Scalability



5 RESULTS (NERSC FS)

STORAGE GROWTH



Term Dist. Over Endpoints

- The majority of terms indexed from filename and pathname metadata appear only on a subset of the endpoints
  - 60.3% of terms exist on one
  - 76.3% of terms exist on two
  - Only 2.1% of terms exist on all
- NERSC FS results are conservative estimates due to similarity
- In general, very few terms will be represented on all endpoints, making the fully distributed model inefficient (very low precision)

SLI Performance

	Lvl 1 Size (GB)	Lvl 1 Size (GB)	Reduction
Sum	39.51	1.08	97.27%
Merge	31.29	0.99	96.83%
Reduction	20.80%	7.78%	

Size reduction is still very high for the NERSC filesystem. Note also that gains from simply merging endpoint-level indexes are only 20.08%, underscoring infeasibility of single central index.

6 CONCLUSIONS

- Our second-level index (SLI) finds a workable compromise between the two traditional large-scale indexing architectures
- The SLI algorithm dramatically reduces the size of a central merged index
  - By using two levels and referencing endpoint-level indexes, the central index can avoid storing all terms, instead only extremely stemmed terms and numerical ranges
  - Because full indexes are retained on endpoints, search is still accurate once redirected to those endpoints
  - Recall stays ~1 because queries at SLI level are stemmed the same as terms. Precision is higher than Traditional Method 2.
- Allows for greater scaling and potentially makes the difference between indexing in memory (fast) and indexing on disk (slow)

7 ACKNOWLEDGEMENTS

This work is supported in part by the National Science Foundation grant NSF-1461260 (BigDataX REU).

9 REFERENCES

1. Foster, Ian. "Globus Online: Accelerating and democratizing science through cloud-based services." IEEE Internet Compu ng 15.3 (2011): 70.