Lucene/Solr 4: A Revolution in Enterprise Search Technology

Search | Discover | Analyze

2013 March 27

Webinar presented by Erik Hatcher

Talk Description

- Lucene/Solr 4 is a ground breaking shift from previous releases. Solr 4.0 dramatically improves scalability, performance, reliability, and flexibility. Lucene 4 has been extensively upgraded. It now supports near real-time (NRT) capabilities that allow indexed documents to be rapidly visible and searchable. Additional Lucene improvements include pluggable scoring, much faster fuzzy and wildcard querying, and vastly improved memory usage.
- The improvements in Lucene have automatically made Solr 4 substantially better. But Solr has also been considerably improved and magnifies these advances with a suite of new “SolrCloud” features that radically improve scalability and reliability.

Takeaways

- What are the Key Feature Enhancements of Lucene/Solr 4, including the new distributed capabilities of SolrCloud
- How to Use the Improved Administrative User Interface
- How Sharding has been improved
- What are the improvements to GeoSpatial Searches, Highlighting, Advanced Query Parsers, Distributed search support, Dynamic core management, Performance statistics, and searches for rare values, such as Primary Key

- What are the Key Feature Enhancements of Lucene/Solr 4, including the new distributed capabilities of SolrCloud?
  - as of Lucene/Solr 4.2(.1)

Key Feature Enhancements of Lucene 4

- Flexible Index Formats
  - New posting list codecs: Block, Simple Text, Append (HDFS..), etc
  - Pulsing codec: improves performance of primary key searches, inlining docs, positions, and payloads, saves disk seeks
- Pluggable Scoring
  - Decoupled from TF/IDF
  - Built in alternatives include BM25 & DFR
    - » http://en.wikipedia.org/wiki/Okapi_BM25
    - » http://terrier.org/docs/v3.5/dfr_description.html
- Faster fuzzy & wildcard query processing
  - Higher performance searching
- String -> BytesRef
  - Much improved data structure
  - … means more less memory and less garbage collection effort

Key Feature Enhancements of Lucene, continued

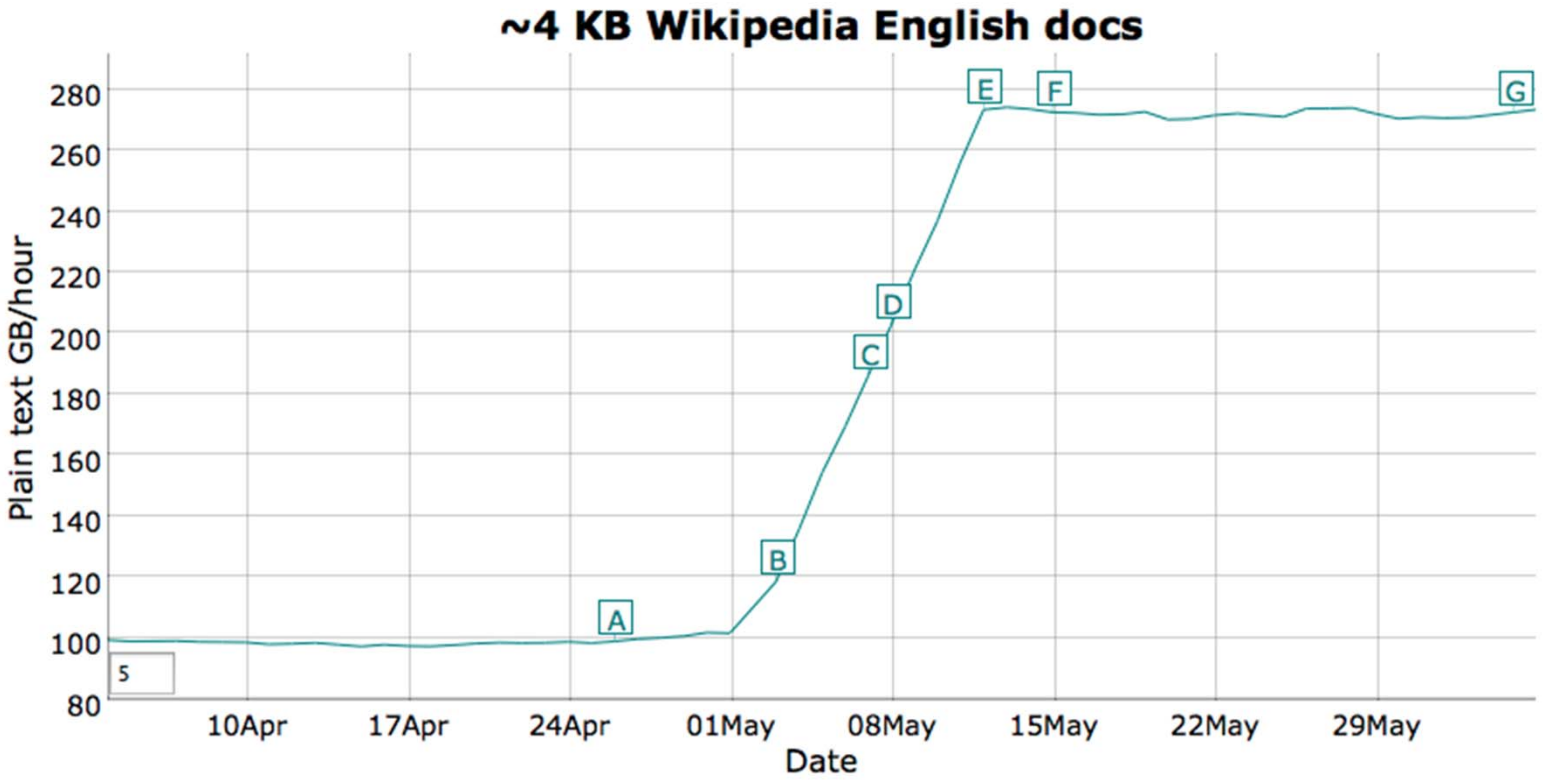
- NRT
  - Per segment
    - » FieldCache can be controlled to only load new segments
    - » Soft commit: faster without fsync, allows quicker update visibility
- DWPT (Document Writer per Thread)
  - Faster more consistent index speed
- DocValues: aka column-stride fields
- DirectSpellChecker
  - Uses main search index directly
- Geospatial Overhaul (covered later)
  - Search in polygonal areas
  - Great for location proximity searching

BytesRef memory management improvements

- On a Wikipedia index (11M documents)
  - Time to perform the first query with sorting (no warmup queries) Solr 3x: 13 seconds, Solr 4: 6 seconds.
  - Memory consumption Solr 3x: 1,040M, Solr 4: 366M. Yes, almost a 2/3 reduction in memory use. And that's the entire program size, not counting memory used to just start Solr and Jetty running.
  - Number of objects on the heap. Solr 3x: 19.4M, Solr 4: 80K. No, that's not a typo. There are over two orders of magnitude fewer objects on the heap in trunk!
  - (data from an Erick Erickson blog entry)

Indexing performance (Wikipedia 4KB docs)

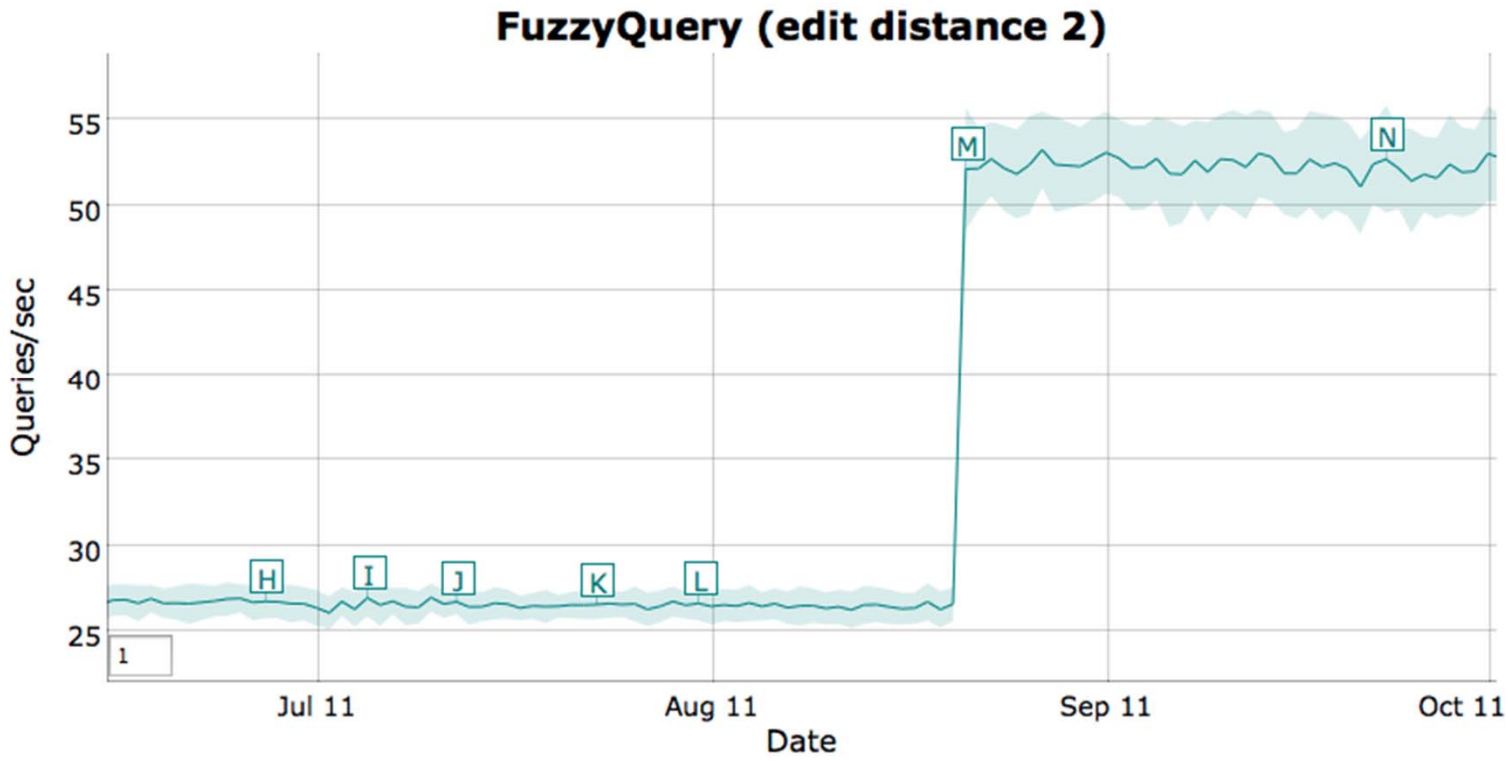
- http://people.apache.org/~mikemccand/lucenebench/indexing.html



LucidWorks™

FuzzyQuery

- http://people.apache.org/~mikemccand/lucenebench/Fuzzy2.html



LucidWorks™

Key Feature Enhancements of Solr 4

- SolrCloud: large, manageable, scaling
- Several new update processors, including a “script” one
- SolrJ streaming response
- Content-type savvy /update handler
- Improved document response: DocTransformer, function calculations
- Pivot faceting
- New relevancy functions
- DirectSpellChecker support
- Pseudo-join
- Improved geospatial capabilities (more detail later)
- NRT/transaction log
- Improved Admin UI
  - including SolrCloud cluster visualizations

SolrCloud

- Distributed/sharded indexing & search
  - Auto distributes updates and queries to appropriate shards
  - Near Real Time (NRT) indexing capable
- Dynamically scalable
  - New SolrCloud instances add indexing and query capacity
- Reliable
  - No single point of failure
  - Transactions logged
  - Robust, automatic recover

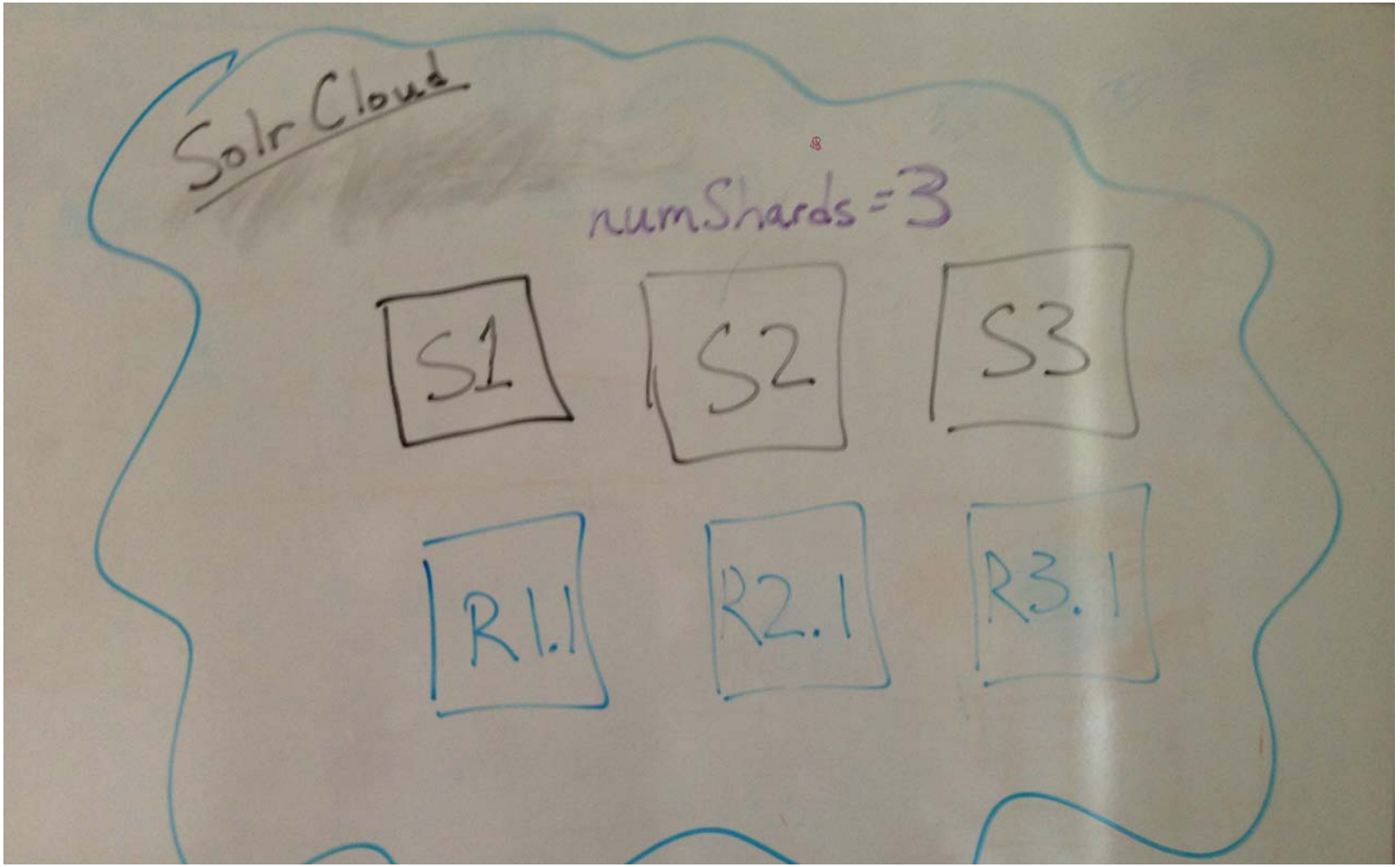
SolrCloud's capabilities

- Transaction log
  - All updates are added to the transaction log. The tlog provides support for: durability for updates that have not yet been committed, peer syncing, real-time get (retrieve documents by unique id) always up to date because it checks the tlog first, does not require opening a new searcher to see changes
- Near Real Time (NRT) indexing
  - Soft commits make updates visible
  - Hard commits make updates durable
- Durability
  - Updates to Solr may be in several different states: buffered in memory, flushed, but not committed or viewable, soft committed (flushed and viewable), committed (durable)
  - The transaction log ensures data is not lost in any of these states if Solr crashes.
- Recovery
  - Solr uses the transaction log for recovery; on startup Solr checks to see if the tlog is in a committed state, if not updates since the last commit are applied
- Optimistic locking
  - Solr maintains a document version (_version_ field); updates can now specify _version_; updates to incorrect version will fail

SolrCloud details

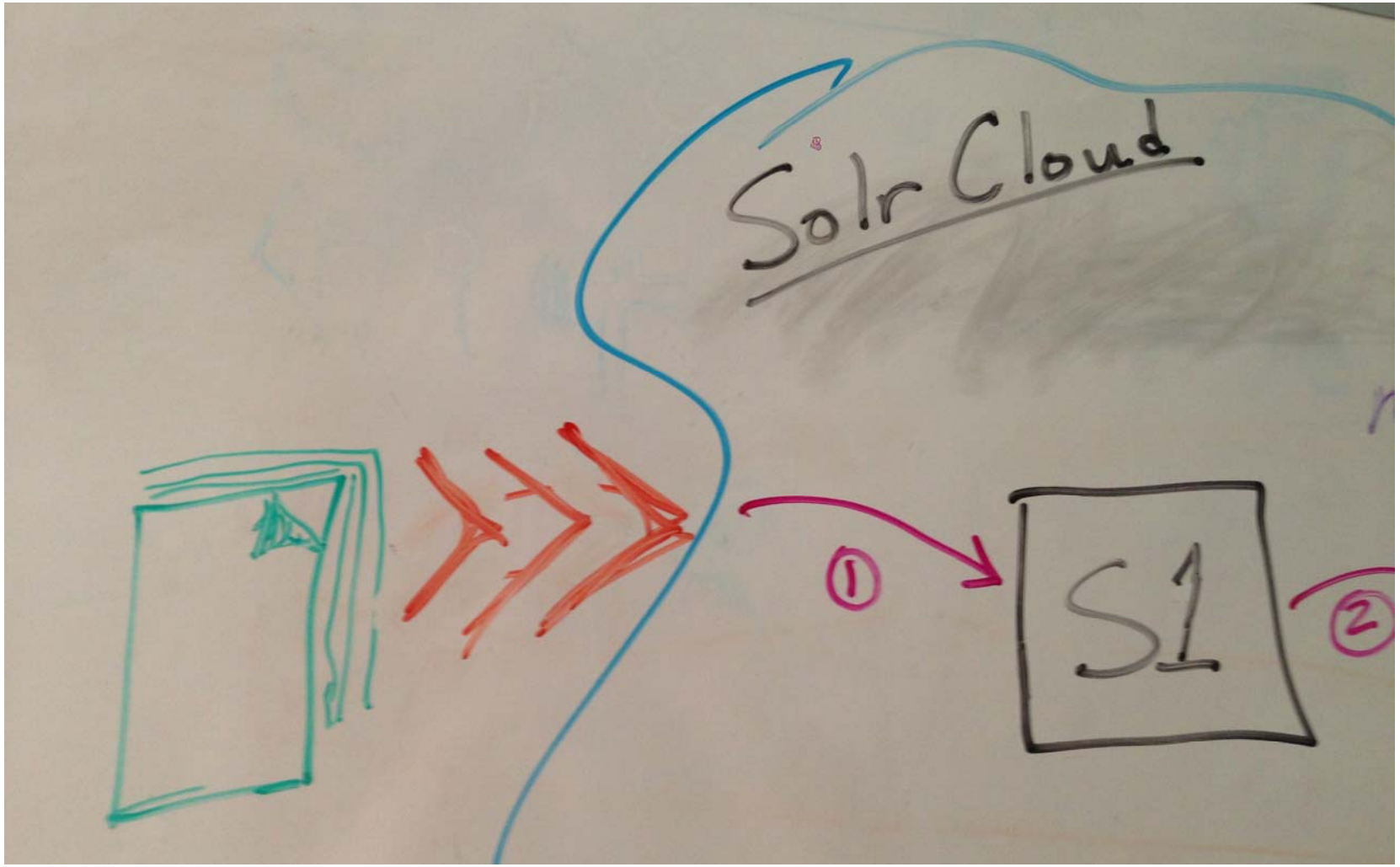
- “Leaders” and “replicas”
  - Leaders are automatically elected
- Leaders are just a replica with some coordination responsibilities for the associated replicas
- If a leader goes down, one of the associated replicas is elected as the new leader
- New nodes are automatically assigned a shard and role, and replicate/recover as needed
- SolrJ’s CloudSolrServer
- Replication in Solr 4
  - Used for new and recovering replicas
  - Or for traditional master/slave configuration

SolrCloud



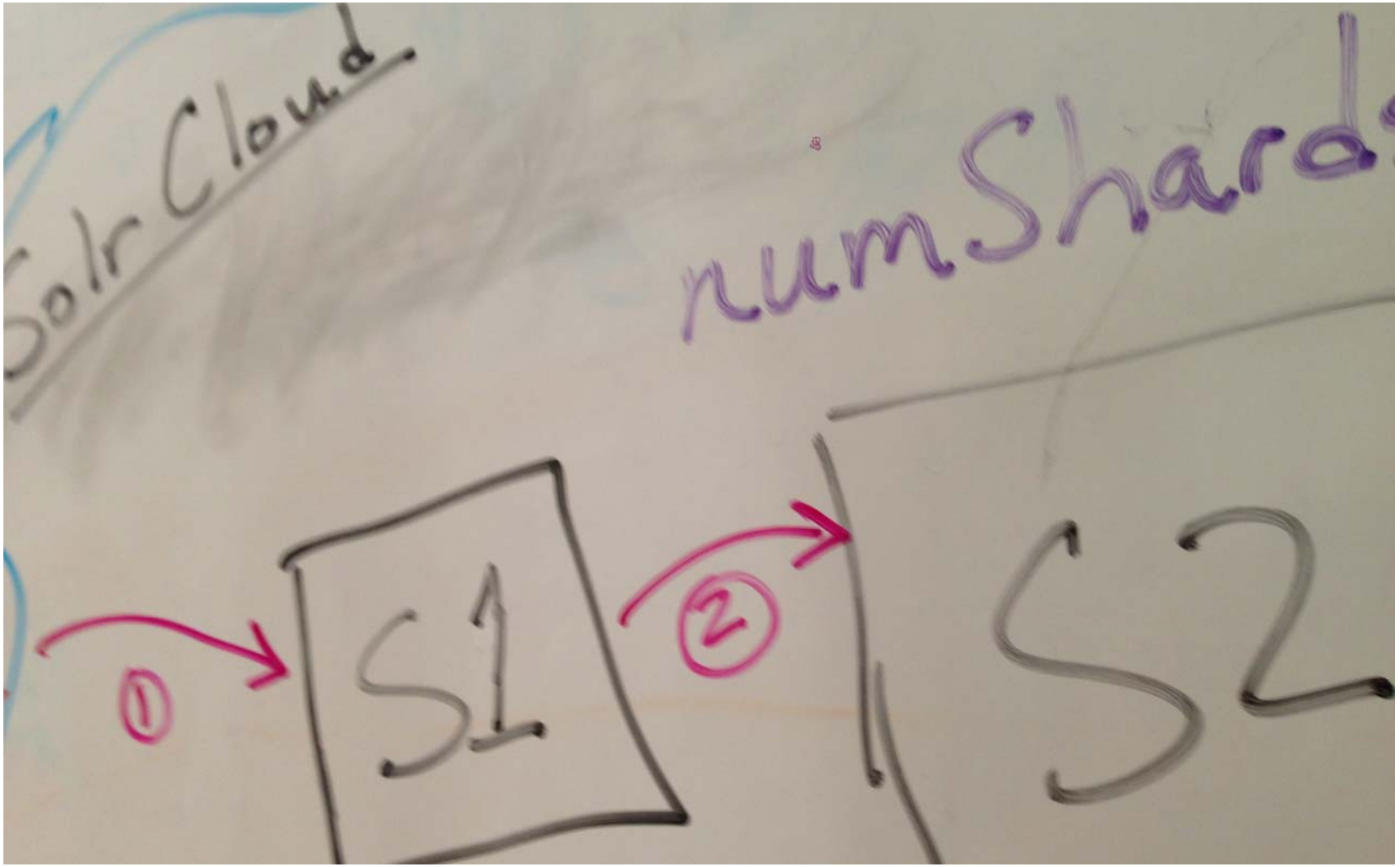
LucidWorks™

Document indexing



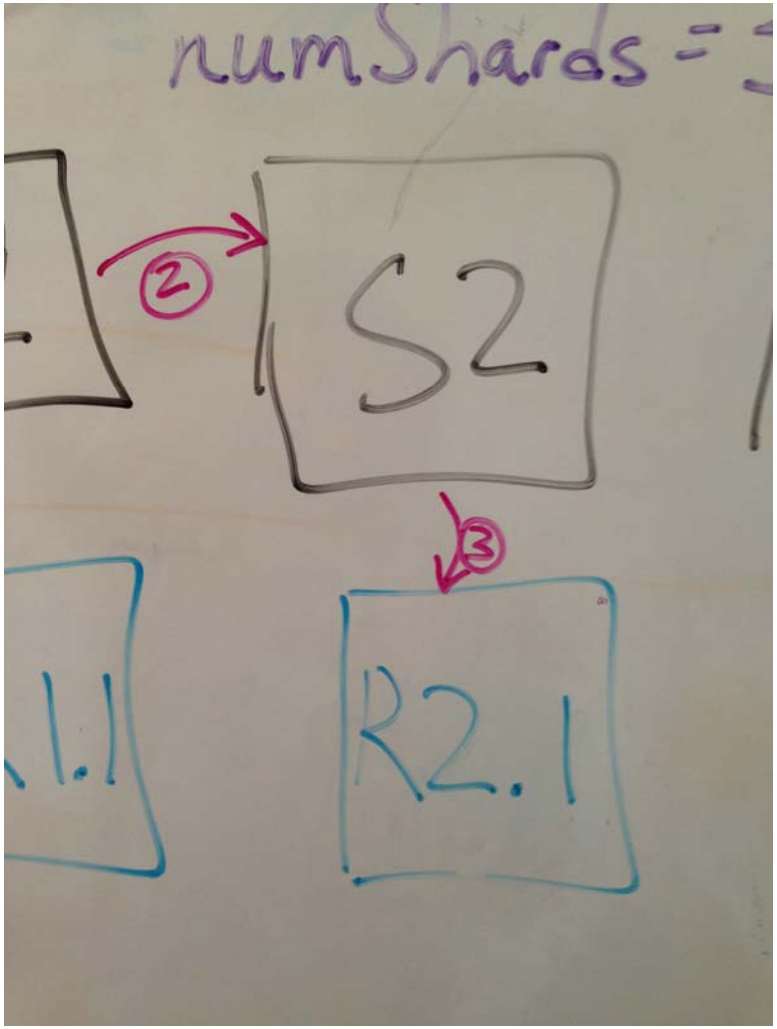
LucidWorks™

Routing to appropriate shard



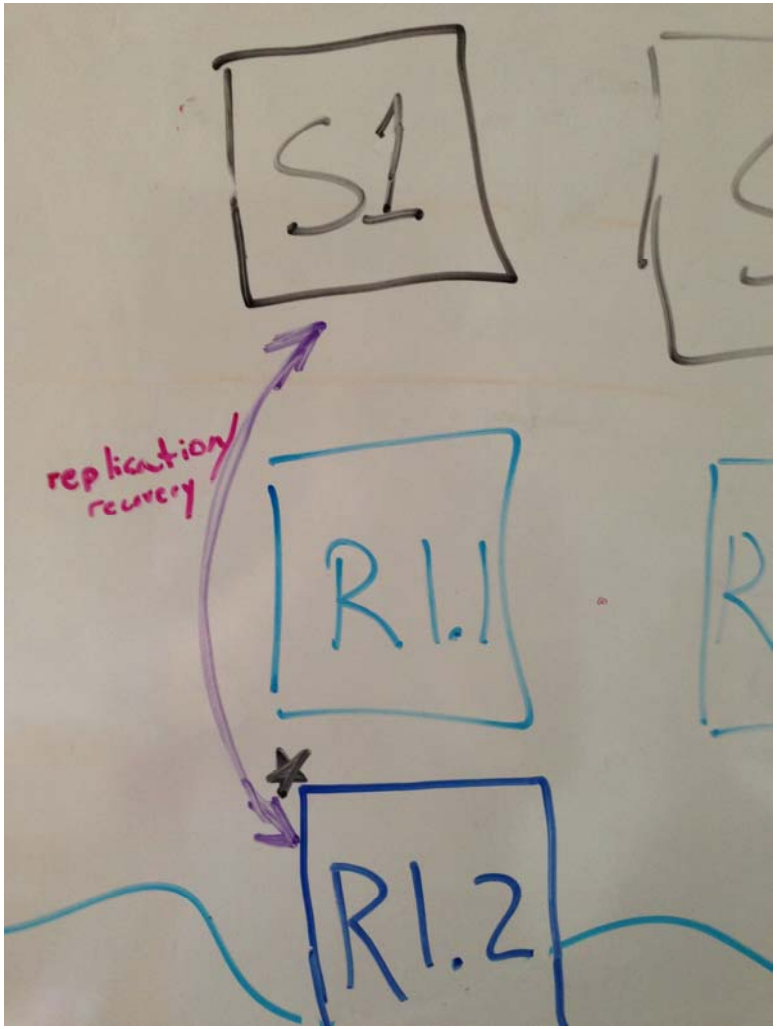
LucidWorks™

Replicating during indexing



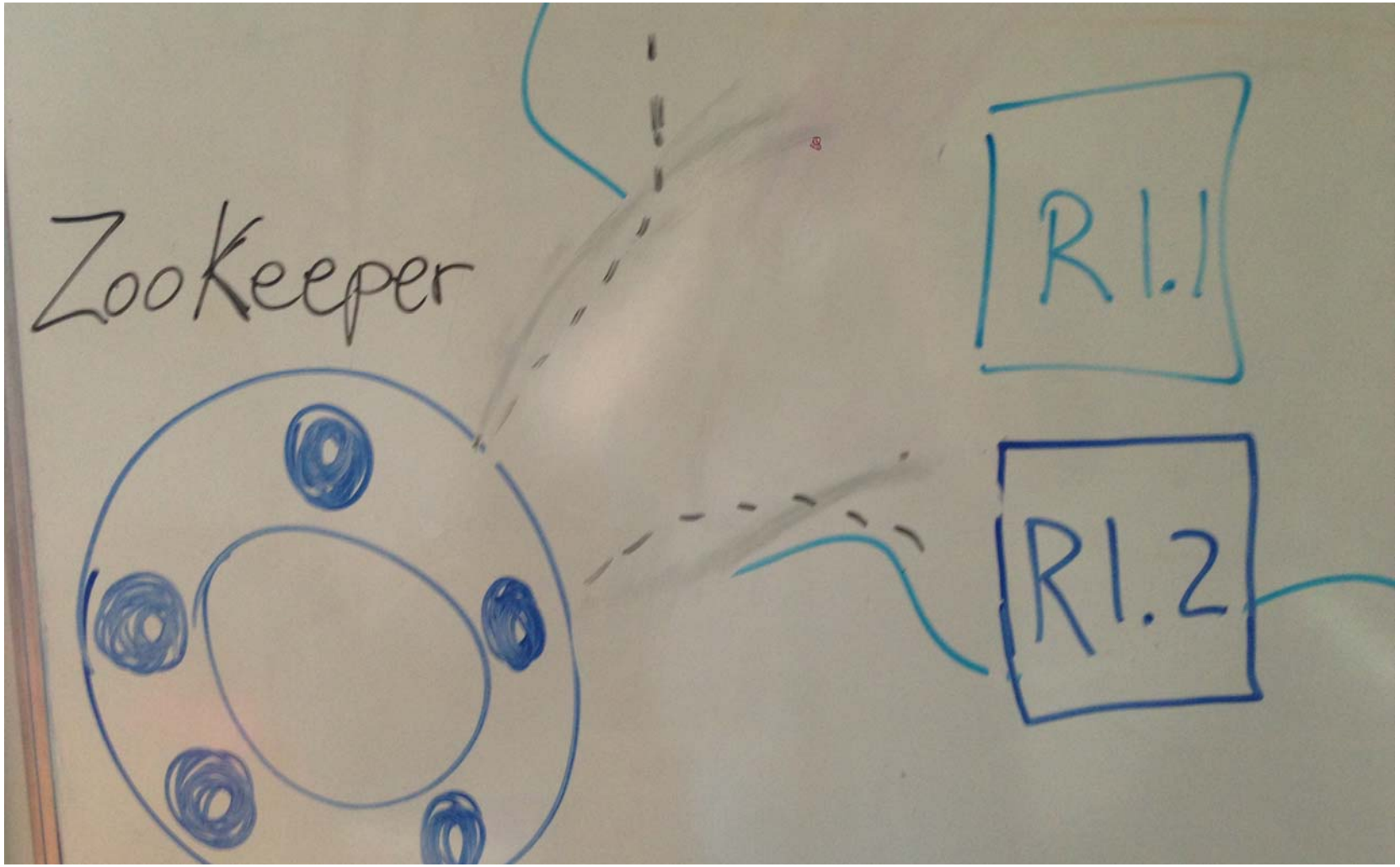
LucidWorks™

Increasing capacity or recovering



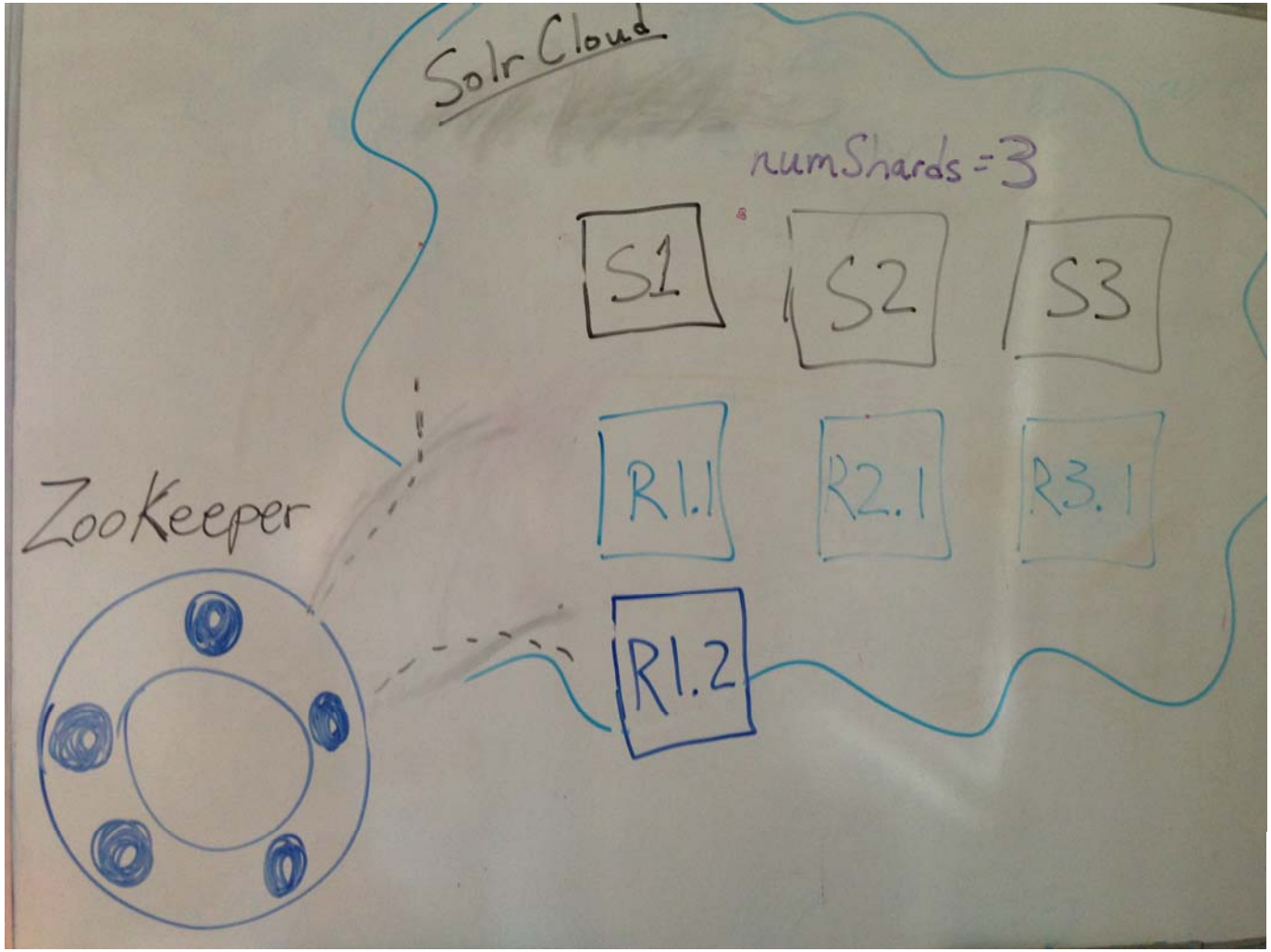
LucidWorks™

ZooKeeper



LucidWorks™

The Big (Data) Picture of SolrCloud



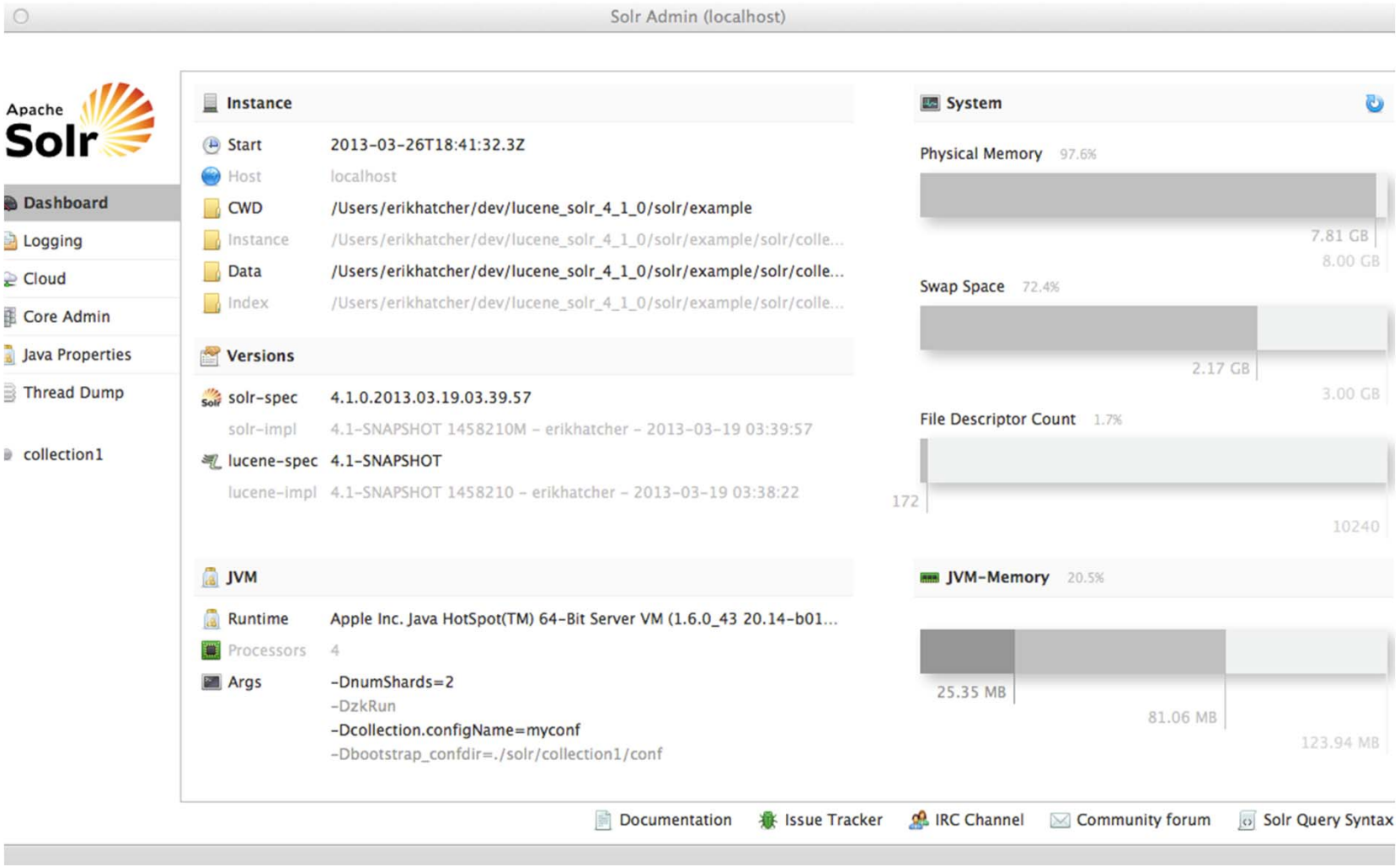
LucidWorks™

- How to Use the Improved Administrative User Interface

Solr Console

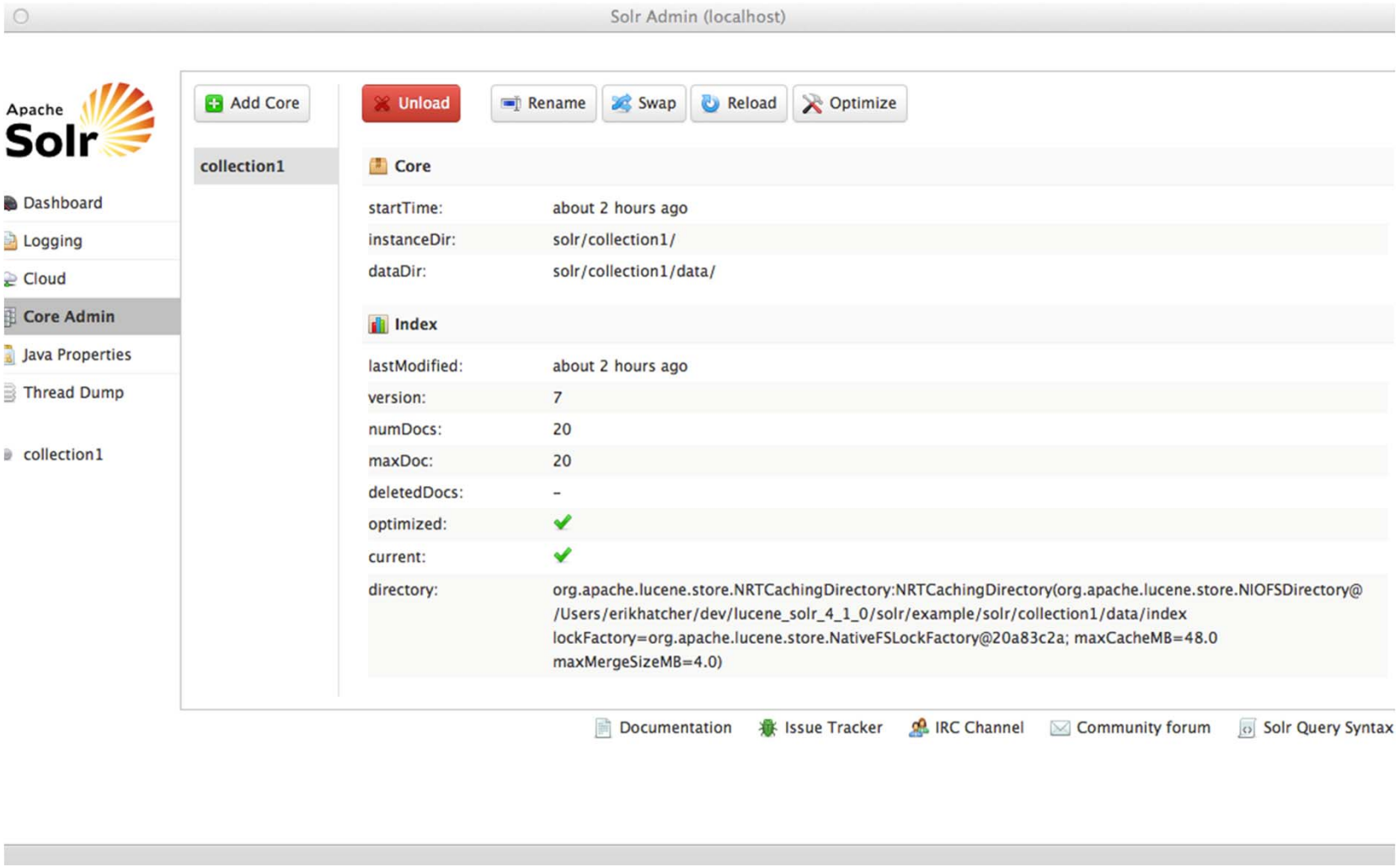
- View SolrCloud cluster state
- View collection schema and configuration
- Querying and analysis tools

/solr/#/



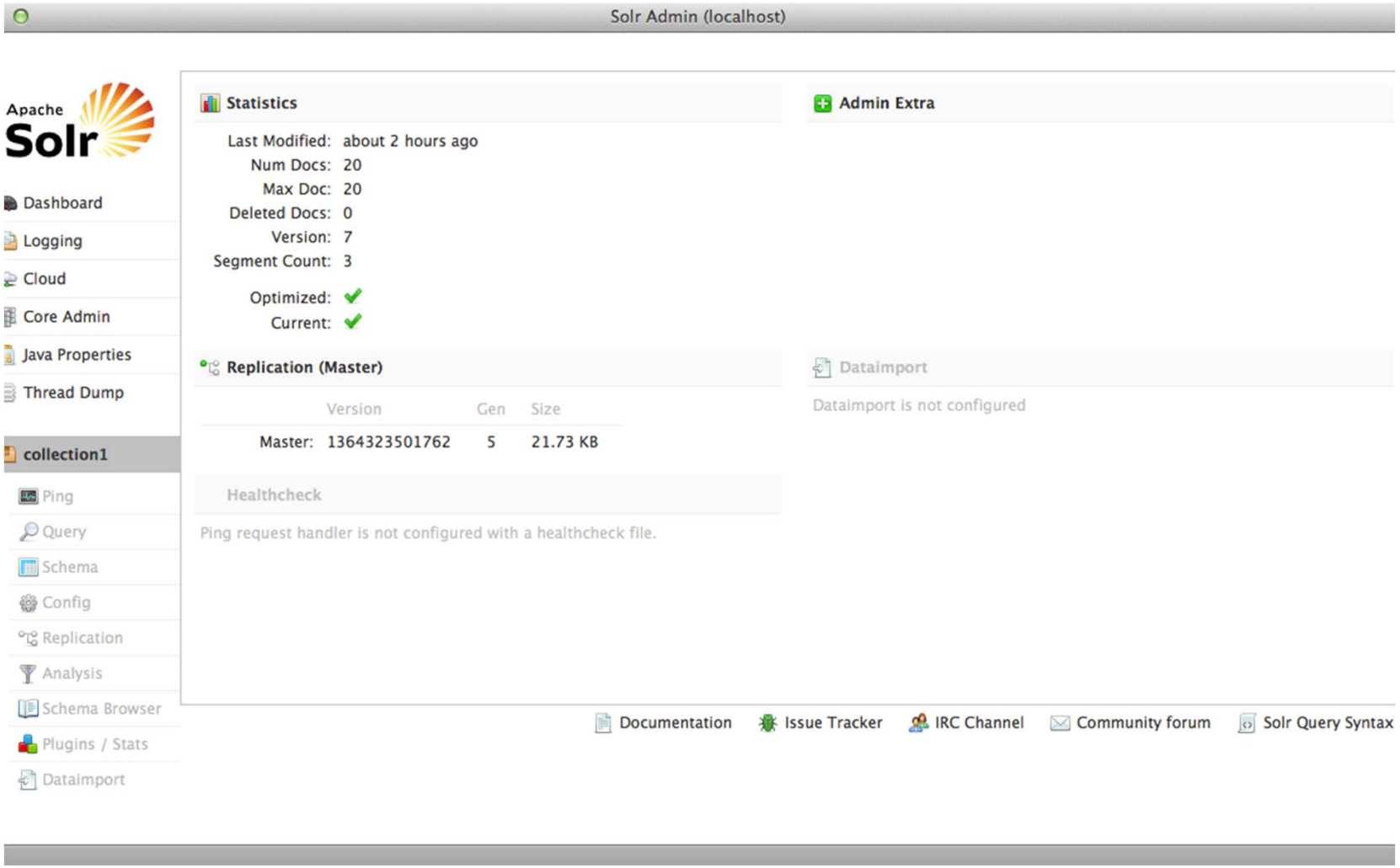
LucidWorks™

Core Admin



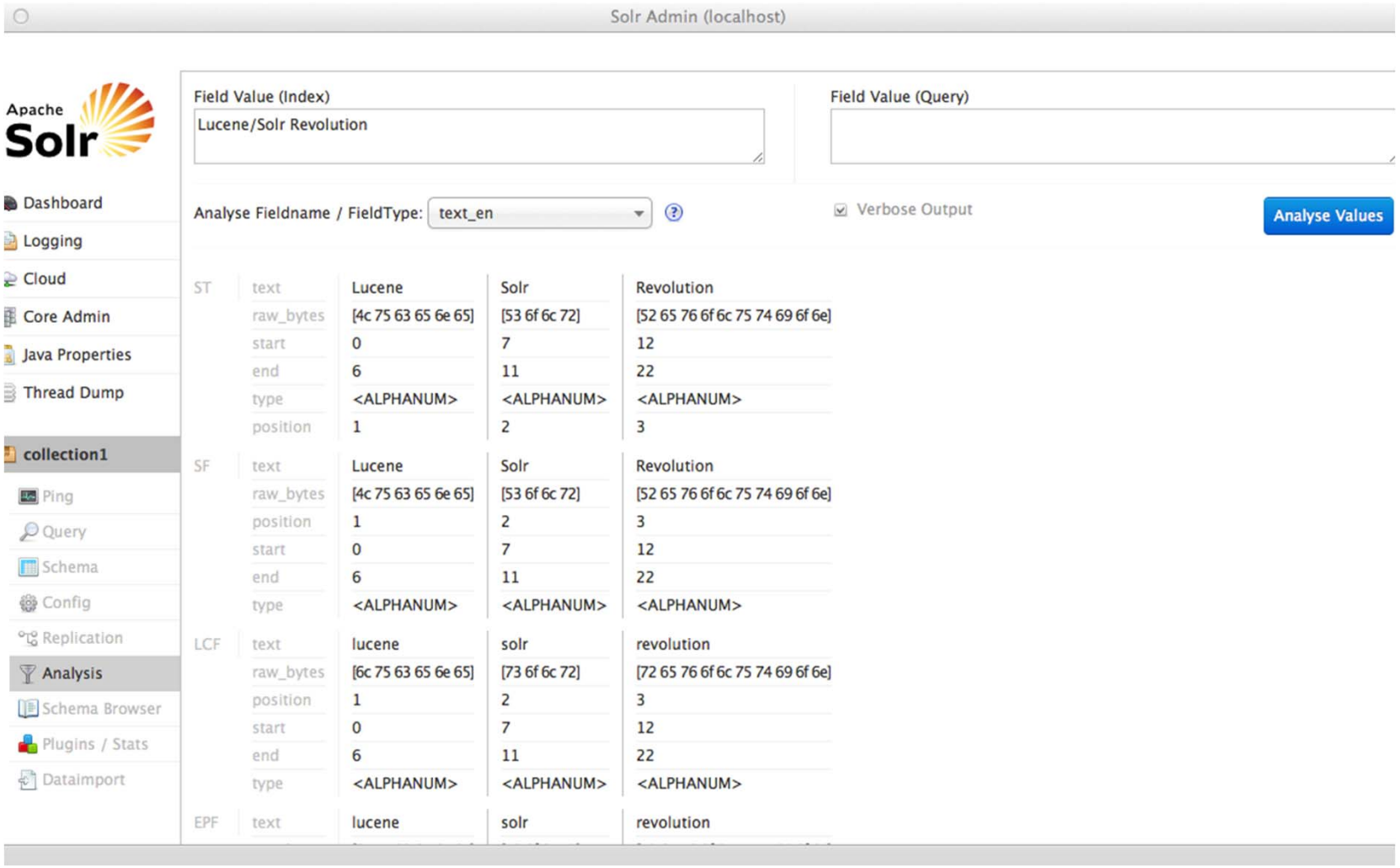
LucidWorks™

Collection View



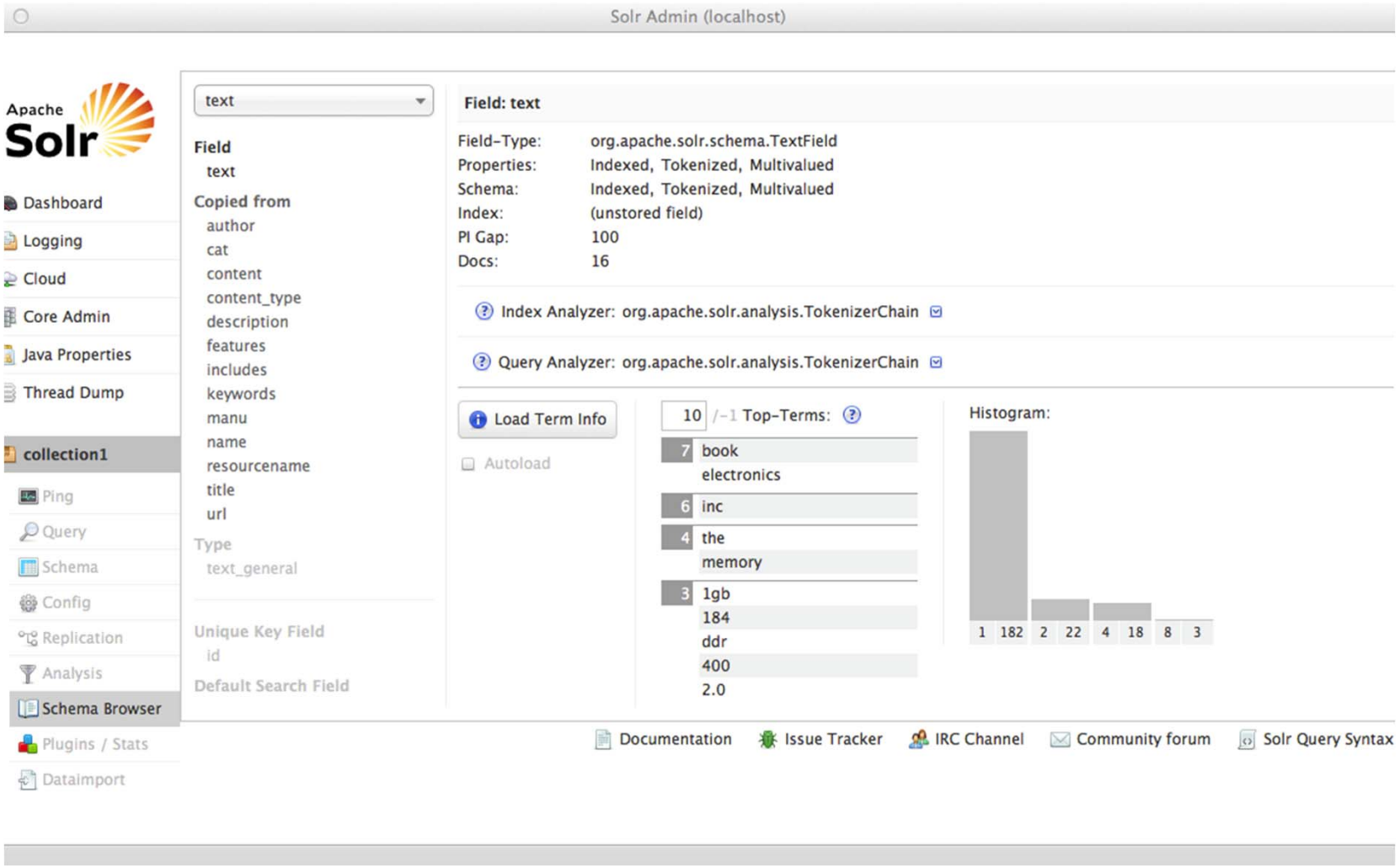
LucidWorks™

Analysis



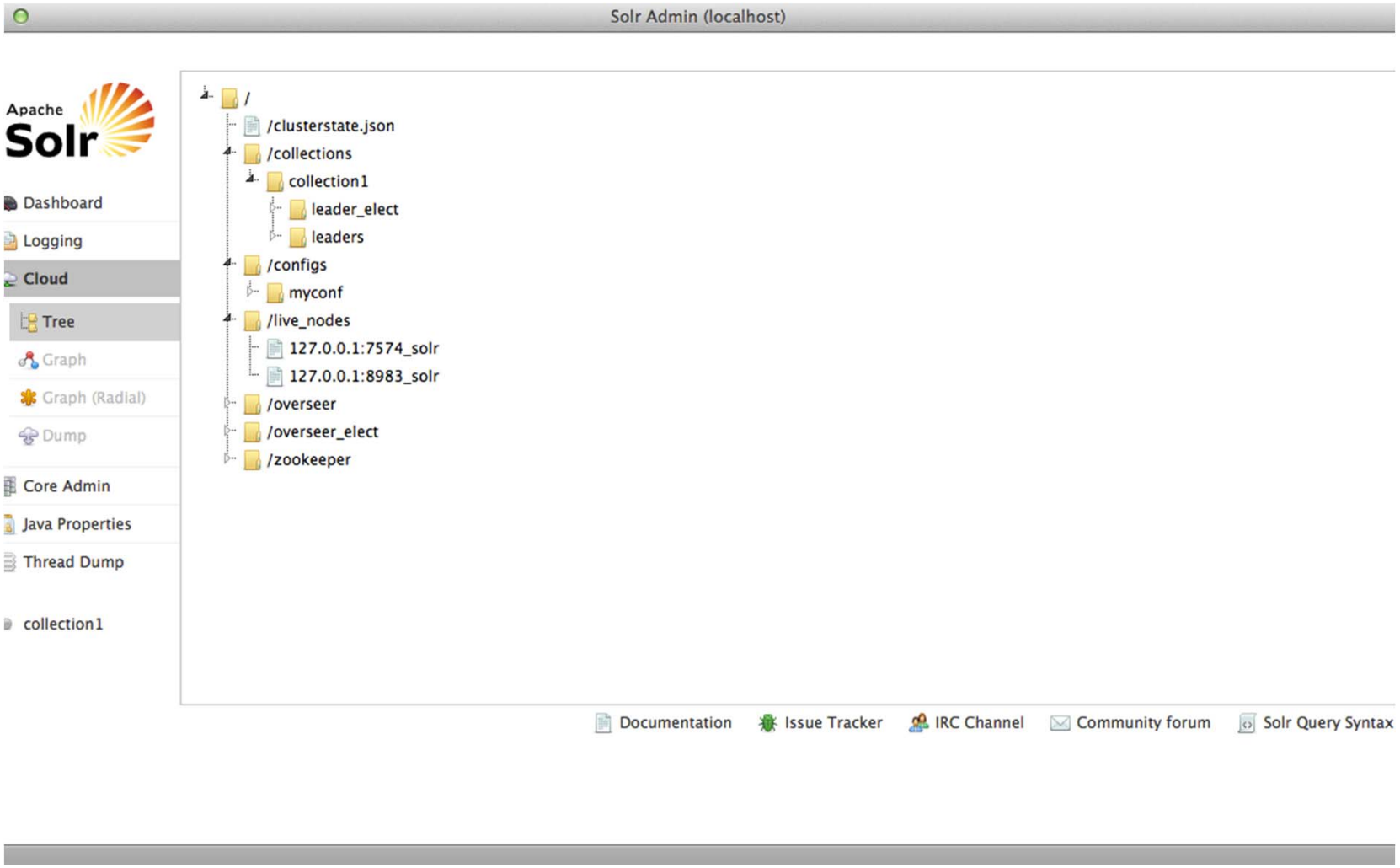
LucidWorks™

Schema Browser



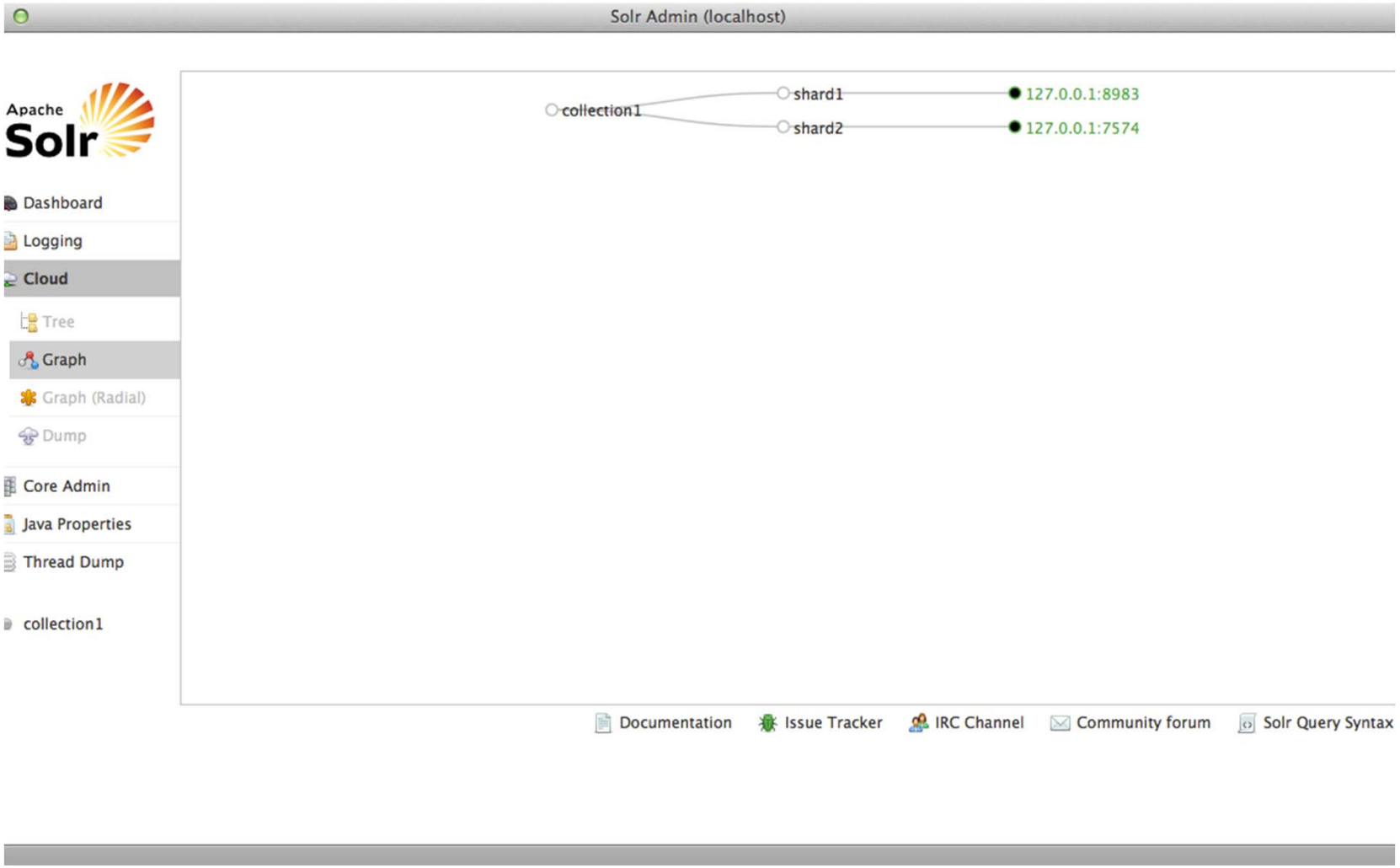
LucidWorks™

SolrCloud: tree view



LucidWorks™

SolrCloud: graph view



LucidWorks™

SolrCloud: radial

LucidWorks™

- How Sharding has been improved

Routing

- Allows you to route documents and queries to a subset of shards
- Provides efficient multi-tenancy
- Indexing:
  - A shard key can be prepended to the unique document id: shard_key!unique_id
  - Documents with the same shard_key will reside on the same shard.
- Querying: shard.keys=shard_key1!...
  - Much more efficient then searching the entire collection.

What are the improvements to … ?

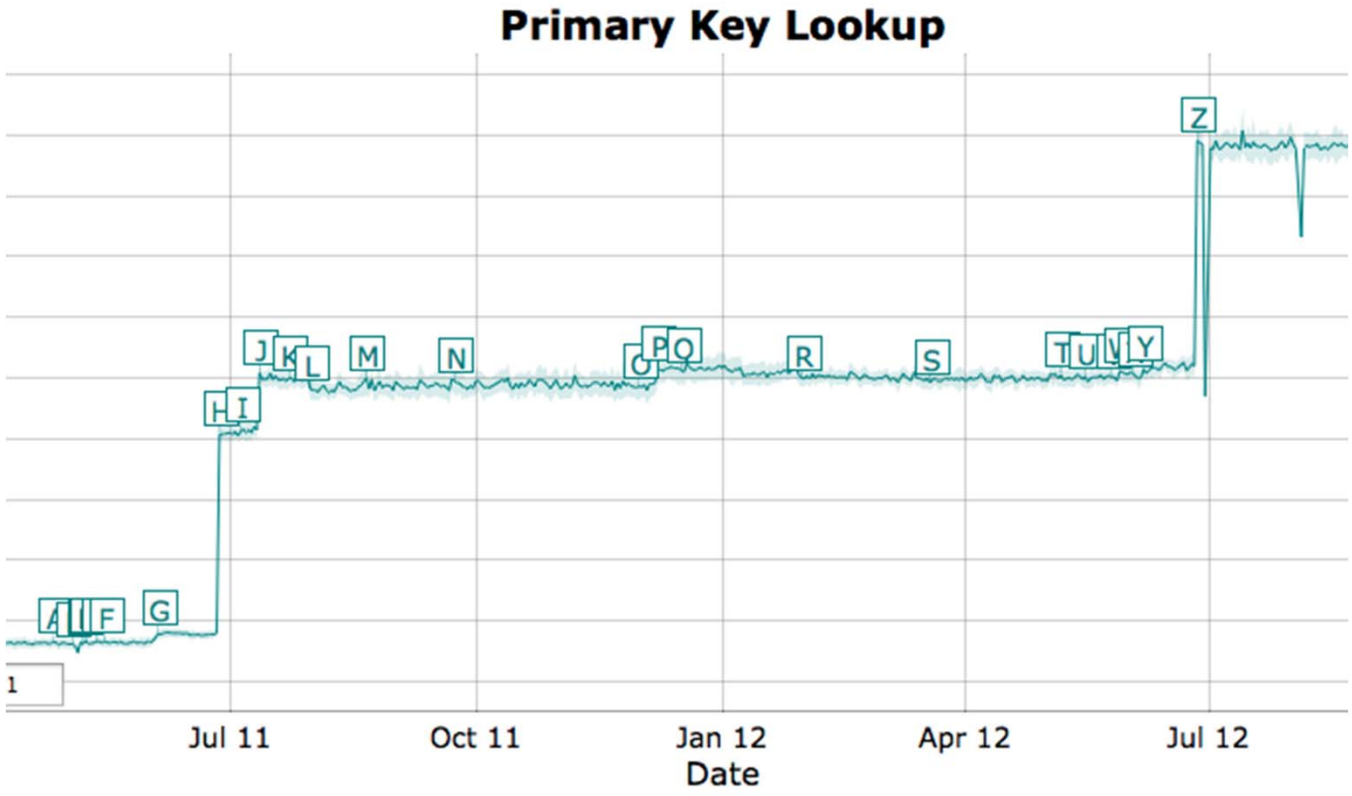
- Geospatial searches
- Highlighting
- Advanced query parsers
- Distributed search support
- Dynamic core management
- Performance statistics
- and searches for rare values, such as Primary Key

Geospatial improvements

- Multiple values per field
- Index shapes other than points (circles, polygons, etc)
- Indexing:
  - "geo”:”43.17614,-90.57341”
  - “geo”:”Circle(4.56,1.23 d=0.0710)”
  - “geo”:”POLYGON((-10 30, -40 40, -10 -20, 40 20, 0 0, -10 30))”
- Searching:
  - fq=geo:"Intersects(-74.093 41.042 -69.347 44.558)"
  - fq=geo:"Intersects(POLYGON((-10 30, -40 40, -10 -20, 40 20, 0 0, -10 30)))"

QPS (primary key lookup)

- http://people.apache.org/~mikemccand/lucenebench/PKLookup.html



LucidWorks™

Solr as NoSQL

- Characteristics
  - Non-traditional data stores
  - Not designed for SQL type queries
  - Distributed fault tolerant architecture
  - Document oriented, data format agnostic(JSON, XML, CSV, binary)
- Updated durability via transaction log
- Real-time /get fetches latest version w/o hard commit
- Versioning and optimistic locking
  - w/ Real Time GET, allows read/write/update w/o conflicts
- Atomic updates
  - Can add/remove/change and increment a field in existing doc w/o re-indexing

Distributed Key / Value Pair Database

- Real-time Get combined with Solr Cloud make a very powerful key/value pair database
  - Durable (tlog)
  - Isolated (Optimistic locking)
  - Redundant (Solr Cloud Replicas)
  - Distributed & scalable (billions of keys, Solr Cloud Sharding)
  - Efficient Multi-tenant (Solr Cloud document routing, Solr 4.1)
  - Fast (milli-second response time, Pulsing Codec)
  - Real-time (tlog)

Looking ahead

- A couple of notable features on the horizon:
  - Automatic shard splitting
  - Query parsing: rich query tree control via JSON/XML
- Continually improving performance, scalability, and robustness

About LucidWorks

- LucidWorks Search
  - Lucene/Solr 4 powered
  - Rich connector framework for SharePoint, web crawling, etc
  - Built-in security support
- LucidWorks Big Data
  - Scalable classification, machine learning, analytics
- Lucene/Solr commercial support
- Consulting
- Training
- http://www.lucidworks.com

Thank You! / Q & A

- Learn More
  - www.SearchHub.org
  - www.LuceneRevolution.org
- Provide Feedback
  - http://svy.mk/10MOIjg
- Join the conversations
  - @LucidWorks
  - @LuceneSolrRev
  - @ErikHatcher

LucidWorks™