Pinky Chauhan
CS 410 Technology Review
University of Illinois at Urbana-Champaign

ElasticSearch vs Solr

Introduction

Elasticsearch and Solr are two of the leading open source search engine technologies. Both are built on top of the same core underlying search library – Lucene and have several similar features as a result, but they are quite different in terms of functionalities such as scalability, ease of deployment, as well as community presence and many other aspects. I will try to highlight these differences and discuss the applicability of these technologies to different use cases in this technology review.

Details

Lucene overview

Apache Lucene is a high-performance, full-featured text search engine library written originally in Java. It is a technology suitable for nearly any application that requires full-text search. It uses inverted index to store the data and provides the following features:

- Scalable, High-Performance Indexing
  - over 150GB/hour on modern hardware
  - small RAM requirements -- only 1MB heap
  - incremental indexing as fast as batch indexing
  - index size roughly 20-30% the size of text indexed
- Powerful, Accurate and Efficient Search Algorithms
  - ranked searching -- best results returned first
  - many powerful query types: phrase queries, wildcard queries, proximity queries, range queries and more
  - fielded searching (e.g. title, author, contents)
  - sorting by any field
  - multiple-index searching with merged results
  - allows simultaneous update and searching
  - flexible faceting, highlighting, joins and result grouping
  - fast, memory-efficient and typo-tolerant suggesters
  - pluggable ranking models, including the Vector Space Model and Okapi BM25

ElasticSearch overview

Elasticsearch is a search engine based on the Lucene library. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents.

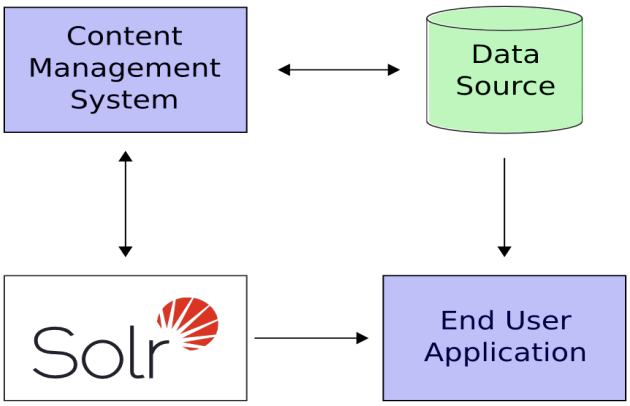
With REST APIs, Elasticsearch leverages on the search and indexing functions of Apache Lucene. This tool also provides a distributed full text search engine along with an HTTP web interface.

Released initially in the year 2010, Elasticsearch is popular for its REST APIs usage, distributed architecture, along with its speed and scalability. Elasticsearch is an integral component of the ELK Stack tools (comprising Elasticsearch, Logstash, and Kibana) – that are used for data ingestion, storage, analysis, and visualization.

Apache Solr overview

Solr is a highly reliable, scalable and fault tolerant open source search engine, providing distributed indexing, replication and load-balanced querying, automated failover and recovery, centralized configuration and more. Solr powers the search and navigation features of many of the world's largest internet sites.

It is built on top of Apache Lucene software library. With HTTP requests, Apache Solr provides advanced search capabilities of Apache Lucene.



Initially released in the year 2004, Apache Solr has a large and growing user community. Some of its best features include distributed full text search, faceting, and real-time indexing. The latest release of Apache Solr is version 8.6 – that was released in July 2020.

As a standalone search server, Solr uses a REST-like API – using which you can index documents in JSON, XML, and CSV formats.

Differences between ElasticSearch and Apache Solr

	SOLR	ELASTICSEARCH
Major features	Mainly focused on text-based searching Highlighting Full-text search Faceted search Real-time indexing Dynamic clustering Database integration NoSQL features Rich document handling	Along with text based search, it is also inclined to log analytics. Multi-tenancy An analyzer chain Analytical search Grouping & aggregation Distributed search
Open source	Fully open source with vast community support	Recently became fully open source with Elasticsearch core and X-Pack (X-Pack code has been released as open source, but still requires commercial licensing to implement)
Initial setup	Although it has extensive documentation, setting up and experimenting on Solr is a bit tedious. Solr is now working on making it more under friendly.	Elastic search is not as rigid as Solr. It's very user friendly and set up is as simple as downloading and executing with a single command.
Queries	Queries can return in JSON, XML and CSV formats	Only JSON format is supported.
Architecture	Solr creates inverted index of the documents posted to its core using a defined schema. Schema is a blue print which helps Solr in creating inverted	Nodes make up a cluster and contain shards, which contain documents that you're searching through. Elasticsearch routes requests through nodes; the nodes

	index of the documents by giving a set of predefined fields. Once Solr completes indexing the documents posted to its core or collection it can be used to run queries. When a query is given to Solr, it breaks the query into different chunks or entities and matches it with the inverted index of the documents created earlier.	then merge results from shards (Lucene indices) together to create a search result. The distributed nature provides redundancy in case of node failures, and also adds capacity in case of heavy traffic.
Scalability	Solr itself is not very scalable but with the help of SolrCloud managed by zookeeper does the job. Solr allows the addition of shards on the go based on the requirements of the applications.	Elasticsearch is scalable and is very flexible when it comes to data clusters. Elastic search actually discourages the practice to add shards on the go. It has a fixed number of primary shards that cannot be altered. This is done to increase the query efficiency as the number of shards/indexes increase, latency in fetching the results increase.
Resource allocation	Solr is static compared to ElasticSearch. Resources are pre-allocated.	Real-time resource allocation and processing with sharding support
Plugins support	Excellent pluggable architecture	More restrictive plugin architecture
	Plugins can be easily developed and integrated	Plugins are not supported in hosted environments
	Tight integration with Lucene development	Lags slightly in implementing new Lucene features

		Frequent point releases with feature additions
Analytics	Strong facet-based analytics	Strong analytic capabilities with aggregations
	JSON facets added to support more dynamic aggregations with analytic functions	Supports analysis on top of aggregations (e.g. moving averages)
	Stream Expressions are added in Solr 7 to support a streaming framework for parallel computation and result emissions for downstream processing	Provides time-series analysis of continually added data (like logs or social media streams) for trend and efficacy insights
Nested Data Structures	Has the notion of parent-child document relationships	Deep nesting is well-supported
	These exist as separate documents within the index, limiting their aggregation functionality in deeply-nested data structures	Fully-structured JSON documents can be directly persisted into Elasticsearch
		Aggregations can be performed against nested structures easily
Query Operations	Mostly limited to query URI parameters, leading to complex queries (debuggable in Solr Admin)	Full-featured Query DSL for writing and expressing complex queries
	JSON API (Solr 7) introduced to allow for JSON based query expressions	Limited to only JSON
	Request handlers can be simply defined in Solr configuration and Java to perform specific and complex tasks related to a given query use case	Custom request handlers require the development of a plugin. There is no notion of jar references from a custom endpoint as there is in Solr
Cognitive Search Capabilities and Integration	Learning to Rank (LTR) module is supported in Solr 6.4 or later	Includes a Machine Learning component (with X-Pack)
	As an Apache project, Solr integrates well with OpenNLP (but not an	

	embedded component) for entity extraction and tagging to feed concept-based search	Allows for pattern recognition and time series forecasting (ML and Kibana) Learning to Rank (LTR) plugin supports machine-learning-driven relevancy tuning exercises Open NLP can be utilized in a similar fashion to Solr as an external component supporting cognitive search functions
Management and Operations	Overall, more difficult to manage (though Cloudera Manager helps with this in a Hadoop environment) APIs are not available (though Solr 7 supports metrics APIs, requires JMX) Scaling requires manual intervention for shard rebalancing (Solr 7 has an auto-scaling API giving some control over shard allocation and distribution)	Easy to set up and scale Automatic shard rebalancing after node addition APIs provide ease of monitoring and state evaluation X-Pack provides out of the box resource dashboards (requires licensing from Elastic)
Bulk Indexing Tools	Batch API operations	Bulk API operations only Configuration modifications can be made to speed up initial bulk indexing
API Interaction	SolrJ (Java) is the most well maintained and up-to-date version and is maintained as part of the Apache project Other Apache maintained APIs: Flare, PHP, Python, Perl Other language APIs exist but are community maintained, and often lag in	Many APIs are developed and supported directly by Elastic (Java, JavaScript, Groovy, .NET, PHP, Perl, Python, Ruby) Other community APIs exist for Elasticsearch (e.g. C++, Erlang, Go, Haskell, Lua, Perl, R, etc.)

	functionality behind SolrJ (most notably the .NET API)	
Use Cases	Search for large bulk data sets, for example, healthcare (payer / provider), biopharma research, finance, and government	Log analytics: enterprise log consumption and analysis or a replacement option for commercial off-the-shelf log analytics products
	Native unformatted record filter and search, such as e-commerce or customer-facing search	Real-time dashboards for operational timeline or sales and marketing insights
	Static data set searching	High-volume data streams with natural language content from social media and IoT streams
	Large bulk reprocessing	Native unformatted record filter and search (e-commerce, customer)

Conclusion

Overall, Solr and Elasticsearch seem similar in working but are different in a few areas like scalability, functionalities, ease of deployment, etc.

While both products are document-oriented search engines, Solr is more focused on enterprise-directed text searches with advanced information retrieval. Consequently, it's more suited for search applications that use massive amounts of static data (for instance e-commerce). Solr fits better into enterprise applications that already implement big data ecosystem tools, such as Hadoop and Spark.

Elasticsearch is focused more on scaling, data analytics, and processing time series data (such as log analysis) to obtain meaningful insights and patterns. Elasticsearch is more suited to modern web applications where data is carried in and out in JSON format.

References:

- ***https://lucene.apache.org/***
- ***https://www.elastic.co/***
- ***https://lucene.apache.org/solr/***
- ***https://mindmajix.com/apache-solr-overview***
- ***https://buildingvts.com/elasticsearch-architectural-overview-a35d3910e515***