Mini-Google Search Engine

Jingwen Qiang, Kai Zhong, Bo Lyu, Shu Yang

May 27, 2020

Part 1. Introduction

It is a stream-based, distributed and highly scalable search-engine implemented in Java, function just as early Google in 2000's but with higher performance.

Division of Labor:
Crawler(Jingwen Qiang)
Indexer/TF-IDF Retrieval Engine(Kai Zhong)
PageRank(Lyu Bo)
Search Engine and User Interface(Shu Yang).

Part 2. Architecture and Implementation

2.1 Crawler

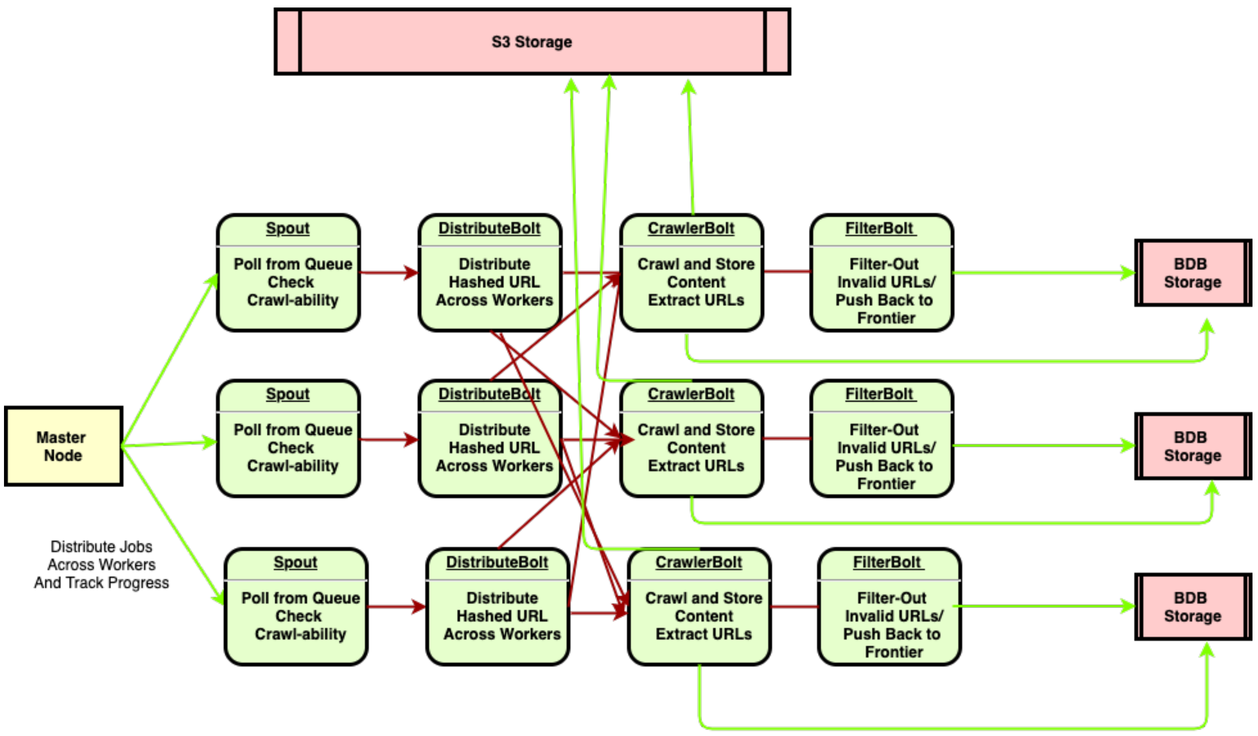
Our Web Crawler is implemented in a (Hadoop liked) stream-processing based distributed manner. We used the stormlite structure to make it event-driven. It is a Mercator-style multi-threaded crawler that can be easily scaled up. The implementation details are the following:

1. **Master-Worker Structure:**
   The overall structure contains a centralized master node that is in charge of the tasks for the workers. with a salable worker system nodes that can be easily instantiated and scaled. Each worker has its own database system in order to optimize the concurrent read and write for databases.

2. **Stream Process: Distributed + Local:**
   In order to optimize the speed, we eliminated the TCP communication between workers for only hashing the designated urls in *Distribute Bolt*. Everything else are handled locally within each worker.

3. **Partitioned and RAM access Database System:**
   Each worker has its own **Berkeley DB** to put data aside from corpus: such as $< url\_id, url >, < url\_id, extracted\_url\_Ids >, < visited\_url, last\_accessed\_time >, < FrontierQueue >$. This increases our data processing speed by having multiple DBs for read and write. In the cloud, we used **Amazon EBS** for the storage for each workers to provide random access.

4. **Save-State Crawling with FrontierQueue Cache:**
   In order to implement multi-threaded crawler as well as save current crawl state for next crawl, we implemented a frontier queue.


S3 Storage
Spout
Poll from Queue Check Crawl-ability
DistributeBolt
Distribute Hashed URL Across Workers
CrawlerBolt
Crawl and Store Content Extract URLs
FilterBolt
Filter-Out Invalid URLs/ Push Back to Frontier
BDB Storage
Master Node
Distribute Jobs Across Workers And Track Progress

2.2 Indexer

The indexer is programmed based on the multi-threaded StormLite Distributed framework, and uses BerkeleyDB to store index data such as inverted index and tf or word location information. BerkeleyDB is good for its random access to read out, so considering the general performance of search engine, we use BerkeleyDB and Amazon EBS to store the index data.

Here are some details about the index data in BerkeleyDB. **1). DocMeta:** also known as HitLists. Each DocMeta indicates a docID,docParam pair to a specific word in that document or Page, and the docParam packs the parameters such as word TF and word locations in this document. The docParam data structure is extensible and easy to apply for more 'fancy features' in this part such as the words occurrence in meta element. **2). IndexEntry:** also known as HitBins. It is a bucket for DocMetas, aggregate all the DocMeta for a specific word. Because of the distributed architecture, each worker or node stores a part of the IndexEntries.

The basic architecture of indexer is shown above. It is based on a multi-threaded StormLite distributed framework. First, spouts take document contents, which are crawled by our crawler, from the S3, and then emit String content to local mappers. In the mappers, once one of them gets a String content, it will begin to parse the document, lemmarize it and count each word's tf values and so on. After that, the mapper will emit DocMetas corresponding to each word in this document. Each node could send http information from mappers to other nodes' reducers by senderBolts. The routing in this phase is field-based, and it will make the DocMetas referring the specific word go to the same node.

Reducers are updating the BerkeleyDB while the indexer is running, and may not crash if error happens. If some nodes go down, the master node remembers the number of documents the indexer has finished, and after we restart the program, it could start somewhere we want. This somehow keeps fault tolerance.

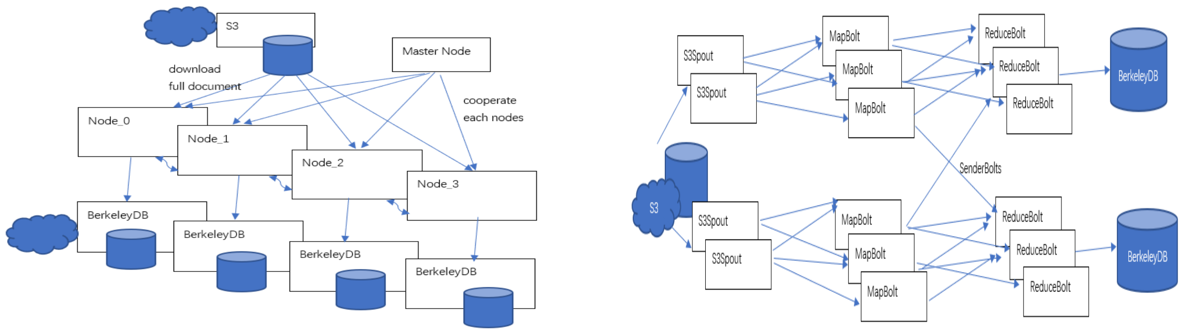


Figure 1: indexer architecture

What we can improve is the running speed. The biggest challenge here is the slow read-write processing time while BerkeleyDB data volume goes up, mostly when document number is above 20k. What we can do is to think out a new storing method in BerkeleyDB, taking place the 'one-by-one' storing process in a reducer.

2.3 PageRank

We implemented the PageRank using StormLite Map Reduce Framework, the result was stored on Berkerley Database. Since the Result was relatively small, we persisted the result on Amazon EBS for random access. To reduce the response time for the query from search engine, how split the the data into different BerkerDB Enviroment based on the hashCode of URL ID.

The Mapper received tuple from the Spout :[Link, Out Link List], and emit tuple [Out Link, Average Rank for Link] to the reducer, and also emit the [Link, Out Link List] to the reducer. Reducer sums up the all Ranks for each Link, and emit [Out Link, New average Rank] and [Link, Out Link List] to the Mapper for next iterations. In the last iterations of Reducer, we write it the Berkery DB.

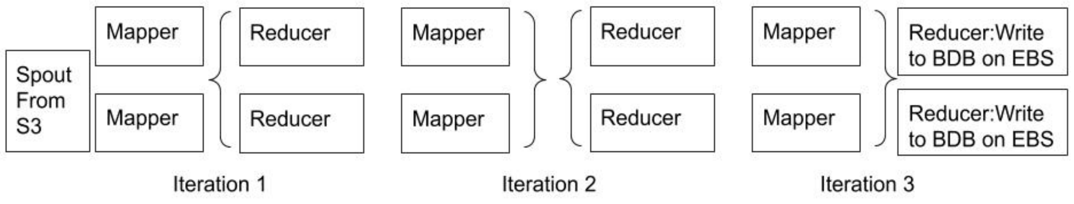


Figure 2: PageRank Architecture

2.4 Search Engine and Web Interface

For searchEngine, google-like spell check, suggest words, Weather api and yelp api. This part focuses more on tuning the algorithm of presenting the results to users. Combined with tfidf and other features, try to get a score and optimize the algorithm to show the top k scores correlated URL, title and its document. In addition, approaching the queries by lemmarizing each keywords. API implementations (weather api and yelp api) are also added to this project.

The web Interface is based on sparkjava, HTML, CSS. Sparkjava providses us with the route, HTML provides the website framework and CSS adds the style to the website. Stanfold core-nlp package was used but is not good as expected so we remove this part to accelerate during the demo.

Open weathermap api and yelp api were used since they are all free of charge and do not need to be processed. They are accessed by key and client ID to finish the tasks assigned. Combined with Bootstrap style, 'card' was directly used. Also the results of the api is merged with queries user input in the input box. When you hit weather, weather api will be presented and when you hit yelp, the yelp api will be present. The default setting of both apis is PA(Philadelphia), but you input other cities or state, it will appear the new one instead. Simple spell check and suggestword are implemented by constructing a big TrieTree Insert the ajax words like into the TrieTree, including every corrected query string. We remove the stopwords to align with the INDEXER's Database.

On tuning the algorithm, Indexer has approached the words in title/ head to be weighted as 3 times than those in the body. Which may improve the final results since users tend to focus on the title and direct notice when seesing the title. They will click on it and see whether the content is a good fit.

Part 3. Evaluation

3.1 Crawler evaluation

	Threads	Pg/Sec/Worker
1	1	2
2	20	8
3	50	10
4	200	10

Table 3.1.1 Crawler Crawling Speed

	Corpus	Pg/Sec/Worker
1	10k	15
2	50k	6-7
3	100k	6-7

Table 3.1.2 Crawler Crawling Speed Over Time

As we see from the diagram above, it shows that there are optimization space in the processing of crawler, which could be frontier queue algorithm or analysis in detail on the crawl time and url filter time.

3.2 Indexer evaluation

For a single worker, which is not distributed, it runs 35.8s for 300 docs, and 3568.2s for 10000 docs, with 100 reducer executors. Generally 1.5 docs/s after 10k docs.
For four workers running a distributed task, it runs 10.4s for 300 docs, and 1522.4s for 10000 docs. with 100 reducer executors each. Generally 5 docs/s after 10k docs.

Although the indexing speed is quite good in the beginning, it becomes much slower after the dataset volume goes up. So we split the 100k docs into 10 pieces to keep the high speed.

3.3 Pagerank evaluation

N(Nodes)	I (Iterations)	K(Workers)	RuningTime
500	6	1	13.1 s
		2	10.7 s
		3	10.8 s
2500	7	1	450.1 s
		2	280.6 s
		3	230.9 s
5000	8	1	620.3 s
		2	420.1 s
		3	259.4 s
10000	15	1	820.1 s
		2	412.3 s
		3	332.0 s
100000	16	1	9720 s
		2	6120 s
		3	4320 s

Table 4.3.1 The Runing Time Evaluation for PageRank

N: The Number of Nodes
I: Number of iterations until convergence
K: Number of workers

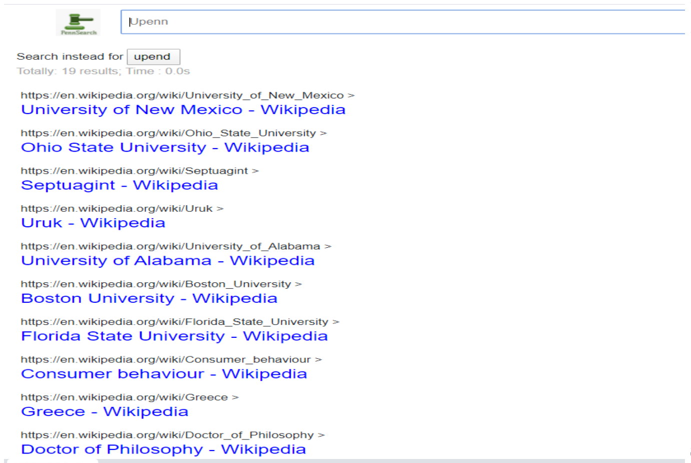
	Number of Query IDs	Response Time
1	1	0.19 s
2	15	0.22 s
3	50	0.23 s
4	100	0.27 s

Table 4.3.2 The Response Time Evaluation for PageRank

3.4 Search engine

Our ranking algorithm is based on the TF-idf parameter and pageRank. The general TF is calculated in the indexer mapper phase, and it is equal to 0.2+[0.2*freq(word,doc)/max(freq(any word,doc))], which is a normalized TF. Furthermore, we emphasize the title words in each document. We take out title word count as a title TF, and add it to the score. So the general score for a document is title_TF*alpha + general_TF + pageRank.

Here are three pictures attached here to show the suggest words function, normal search function and api function.



Upenn

Search instead for upend
Totally: 19 results; Time : 0.0s

https://en.wikipedia.org/wiki/University_of_New_Mexico >
University of New Mexico - Wikipedia

https://en.wikipedia.org/wiki/Ohio_State_University >
Ohio State University - Wikipedia

https://en.wikipedia.org/wiki/Septuagint >
Septuagint - Wikipedia

https://en.wikipedia.org/wiki/Uruk >
Uruk - Wikipedia

https://en.wikipedia.org/wiki/University_of_Alabama >
University of Alabama - Wikipedia

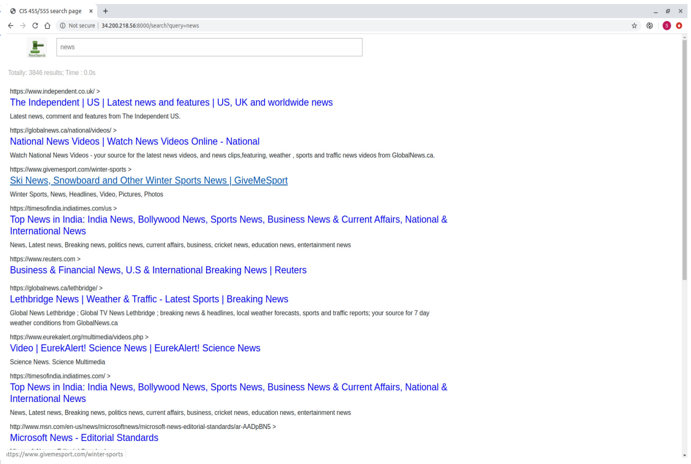
https://en.wikipedia.org/wiki/Boston_University >
Boston University - Wikipedia

https://en.wikipedia.org/wiki/Florida_State_University >
Florida State University - Wikipedia

https://en.wikipedia.org/wiki/Consumer_behaviour >
Consumer behaviour - Wikipedia

https://en.wikipedia.org/wiki/Greece >
Greece - Wikipedia

https://en.wikipedia.org/wiki/Doctor_of_Philosophy >
Doctor of Philosophy - Wikipedia



CIS 455/555 search page
Not secure | 34.200.218.56:8000/search?query=news

news

Totally: 3846 results; Time : 0.0s

https://www.independent.co.uk/ >
The Independent | US | Latest news and features | US, UK and worldwide news
Latest news, comment and features from The Independent US.

https://globalnews.ca/national/videos/ >
National News Videos | Watch News Videos Online - National
Watch National News Videos - your source for the latest news videos, and news clips,featuring, weather , sports and traffic news videos from GlobalNews.ca.

https://www.givemesport.com/winter-sports >
Ski News, Snowboard and Other Winter Sports News | GiveMeSport
Winter Sports, News, Headlines, Video, Pictures, Photos

https://timesofindia.indiatimes.com/us >
Top News in India: India News, Bollywood News, Sports News, Business News & Current Affairs, National & International News
News, Latest news, Breaking news, politics news, current affairs, business, cricket news, education news, entertainment news

https://www.reuters.com >
Business & Financial News, U.S & International Breaking News | Reuters

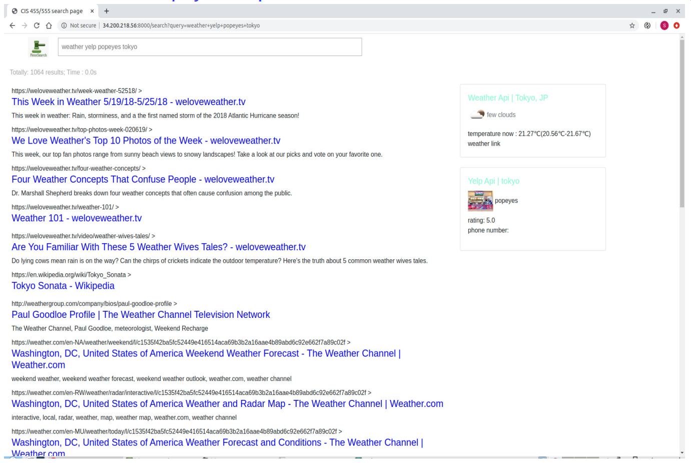
https://globalnews.ca/lethbridge/ >
Lethbridge News | Weather & Traffic - Latest Sports | Breaking News
Global News Lethbridge ; Global TV News Lethbridge ; breaking news & headlines, local weather forecasts, sports and traffic reports; your source for 7 day weather conditions from GlobalNews.ca

https://www.eurekalert.org/multimedia/videos.php >
Video | EurekAlert! Science News | EurekAlert! Science News
Science News. Science Multimedia

https://timesofindia.indiatimes.com/ >
Top News in India: India News, Bollywood News, Sports News, Business News & Current Affairs, National & International News
News, Latest news, Breaking news, politics news, current affairs, business, cricket news, education news, entertainment news

http://www.msn.com/en-us/news/microsoftnews/microsoft-news-editorial-standards/ar-AADpBN5 >
Microsoft News - Editorial Standards

https://www.givemesport.com/winter-sports



CIS 455/555 search page
Not secure | 34.200.218.56:8000/search?query=weather+yelp+popeyes+tokyo

weather yelp popeyes tokyo

Totally: 1064 results; Time : 0.0s

https://weloveweather.tv/week-weather-52518/ >
This Week in Weather 5/19/18-5/25/18 - weloveweather.tv
This week in weather: Rain, storminess, and a the first named storm of the 2018 Atlantic Hurricane season!

https://weloveweather.tv/top-photos-week-020619/ >
We Love Weather's Top 10 Photos of the Week - weloveweather.tv
This week, our top fan photos range from sunny beach views to snowy landscapes! Take a look at our picks and vote on your favorite one.

https://weloveweather.tv/four-weather-concepts/ >
Four Weather Concepts That Confuse People - weloveweather.tv
Dr. Marshall Shepherd breaks down four weather concepts that often cause confusion among the public.

https://weloveweather.tv/weather-101/ >
Weather 101 - weloveweather.tv

https://weloveweather.tv/video/weather-wives-tales/ >
Are You Familiar With These 5 Weather Wives Tales? - weloveweather.tv
Do lying cows mean rain is on the way? Can the chirps of crickets indicate the outdoor temperature? Here's the truth about 5 common weather wives tales.

https://en.wikipedia.org/wiki/Tokyo_Sonata >
Tokyo Sonata - Wikipedia

http://weathergroup.com/company/bios/paul-goodloe-profile >
Paul Goodloe Profile | The Weather Channel Television Network
The Weather Channel, Paul Goodloe, meteorologist, Weekend Recharge

https://weather.com/en-NA/weather/weekend/l/c1535f42ba5fc52449e416514aca69b3b2a16aae4b89abd6c92e662f7a89c02f >
Washington, DC, United States of America Weekend Weather Forecast - The Weather Channel | Weather.com
weekend weather, weekend weather forecast, weekend weather outlook, weather.com, weather channel

https://weather.com/en-RW/weather/radar/interactive/l/c1535f42ba5fc52449e416514aca69b3b2a16aae4b89abd6c92e662f7a89c02f >
Washington, DC, United States of America Weather and Radar Map - The Weather Channel | Weather.com
interactive, local, radar, weather, map, weather map, weather.com, weather channel

https://weather.com/en-MU/weather/today/l/c1535f42ba5fc52449e416514aca69b3b2a16aae4b89abd6c92e662f7a89c02f >
Washington, DC, United States of America Weather Forecast and Conditions - The Weather Channel | Weather.com

Weather Api | Tokyo, JP
few clouds
temperature now : 21.27°C(20.56°C-21.67°C)
weather link

Yelp Api | tokyo
popeyes
rating: 5.0
phone number: