SPERG: Scalable Political Event Report Geoparsing in Big Data

Aswin Krishna Gunasekaran
Computer Science Department
The University of Texas at Dallas
Richardson, Texas
aswin.krishnagunasekaran@utdallas.edu

Maryam Bahojb Imani
Computer Science Department
The University of Texas at Dallas
Richardson, Texas
maryam.imani@utdallas.edu

Latifur Khan
Computer Science Department
The University of Texas at Dallas
Richardson, Texas
lkhan@utdallas.edu

Christan Grant
School of Computer Science
University of Oklahoma
Norman, Oklahoma
cgrant@ou.edu

Patrick T. Brandt
School of Economic, Political and Policy Sciences
The University of Texas at Dallas
Richardson, Texas
pbrandt@utdallas.edu

Jennifer S. Holmes
School of Economic, Political and Policy Sciences
The University of Texas at Dallas
Richardson, Texas
jholmes@utdallas.edu

***Abstract*—Digital newspaper archives accumulated over the last few decades serve as an easily accessible, rich source of information for researchers to conduct analytic studies. Extracting unambiguous geographic identifiers, such as the geographic coordinates solely from text descriptions, a process also known as geoparsing, has proven to be a major challenge when applied to massive corpora like newspaper archives. We focus primarily on archived newspaper reports on political events and aim to parse the exact event location with high accuracy. We identify all the focus locations, which includes all locations mentioned in a report along with the coordinates and task ourselves with recognizing the location where the event actually occurred which we define as our *primary focus location*. Our objective is to extract the latitude-longitude information of these primary focus locations. Existing geoparsers only partially serve the purpose and are not robust enough to process large data archives in reasonable time. In this paper we propose a framework to extract geolocation information of primary focus location from 76 million documents in a distributed environment.**

***Index Terms*—Focus Location Extraction; Distributed Computing; Political Event News;**

I. Introduction

Enormous amounts of political event reports have been stored away for processing over a long period of time. Processing these reports manually is not feasible because of the cost and effort required to do so. Data mining on unstructured data [1] is still a major area of research. Political scientists require information from these reports for various study purposes, and are particularly interested in knowing about event objectives and impact, attendee profile, location of the event, etc. In this paper, we focus on precisely identifying location from this archive.

News reports may have multiple locations mentioned in them, which we call focus locations, and the event it describes

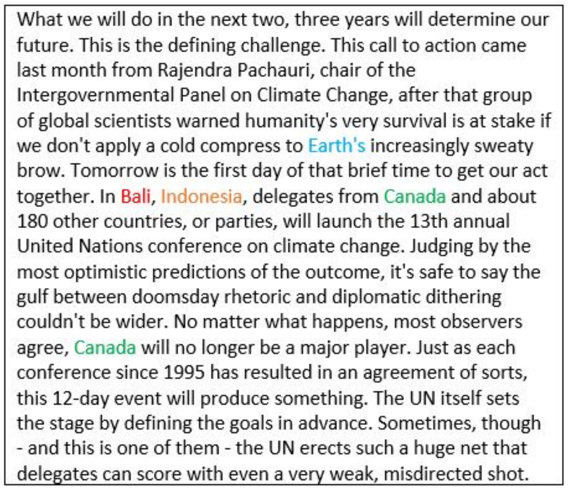
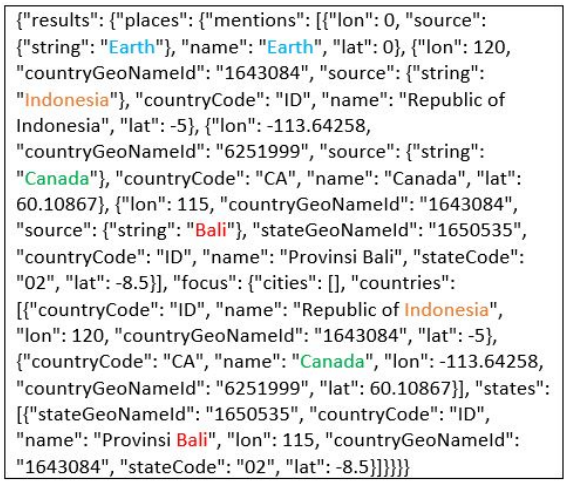

What we will do in the next two, three years will determine our future. This is the defining challenge. This call to action came last month from Rajendra Pachauri, chair of the Intergovernmental Panel on Climate Change, after that group of global scientists warned humanity's very survival is at stake if we don't apply a cold compress to Earth's increasingly sweaty brow. Tomorrow is the first day of that brief time to get our act together. In Bali, Indonesia, delegates from Canada and about 180 other countries, or parties, will launch the 13th annual United Nations conference on climate change. Judging by the most optimistic predictions of the outcome, it's safe to say the gulf between doomsday rhetoric and diplomatic dithering couldn't be wider. No matter what happens, most observers agree, Canada will no longer be a major player. Just as each conference since 1995 has resulted in an agreement of sorts, this 12-day event will produce something. The UN itself sets the stage by defining the goals in advance. Sometimes, though - and this is one of them - the UN erects such a huge net that delegates can score with even a very weak, misdirected shot.

Fig. 1: A sample news report in English

TABLE I: Capabilities of different geoparsers in focus location and primary focus location extraction. Here, ✓ indicates presence and ✗ indicates absence of corresponding capability.

Geoparsers	NER Extraction	Coordinates Extraction	Focus Location	Primary Focus Location	Multi-Lingual
Cliff-Clavin	Stanford CoreNLP	✓	✓	✗	✗
Mordecai	MITIE	✓	✗	✗	✗
Profile	MITIE	✗	✓	✓	✗
Profile (modified)	MITIE	✗	✓	✓	✓

is assumed to be held at a single location, which we define to be our Primary Focus Location. For instance, Fig. 1 is a news report with multiple focus locations mentioned in it such as “Earth”, “Bali”, “Indonesia”, “Canada” but the event is to happen at “Bali” which is the primary focus location.



```
{"results": {"places": {"mentions": [{"lon": 0, "source":
{"string": "Earth"}, "name": "Earth", "lat": 0}, {"lon": 120,
"countryGeoNameId": "1643084", "source": {"string":
"Indonesia"}, "countryCode": "ID", "name": "Republic of
Indonesia", "lat": -5}, {"lon": -113.64258,
"countryGeoNameId": "6251999", "source": {"string":
"Canada"}, "countryCode": "CA", "name": "Canada", "lat":
60.10867}, {"lon": 115, "countryGeoNameId": "1643084",
"source": {"string": "Bali"}, "stateGeoNameId": "1650535",
"countryCode": "ID", "name": "Provinsi Bali", "stateCode":
"02", "lat": -8.5}], "focus": {"cities": [], "countries":
[{"countryCode": "ID", "name": "Republic of Indonesia",
"lon": 120, "countryGeoNameId": "1643084", "lat": -5},
{"countryCode": "CA", "name": "Canada", "lon": -113.64258,
"countryGeoNameId": "6251999", "lat": 60.10867}], "states":
[{"stateGeoNameId": "1650535", "countryCode": "ID",
"name": "Provinsi Bali", "lon": 115, "countryGeoNameId":
"1643084", "stateCode": "02", "lat": -8.5}]}}}}
```

Fig. 2: Output of Cliff-Clavin for the sample news report

Identifying the primary focus location from this text and finding its geo-coordinates is trivial. However, performing the same for millions of documents will span months or even a year, despite using some of the best available geoparsers such as Cliff-Clavin [2], Profile [3] and Mordecai [4]. Existing open source geoparsers either identify the primary focus location or identify all locations along with their coordinates, use different NERs (Named Entity Recognition) and some of them even support multiple languages as depicted in Table I. To our best knowledge, none of the available systems extract coordinates for the primary focus location of an event from a news report.

Using the capabilities of the existing geoparsers, we propose a distributed federated system named Sperg which distributes the workload across several processes utilizing fault tolerant applications to process news reports in a robust fashion. It is fine-tuned to process a million documents within hours and can store the geo-coordinates of the primary focus location in MongoDB. Stored results are later indexed and available for querying to political scientists for research.

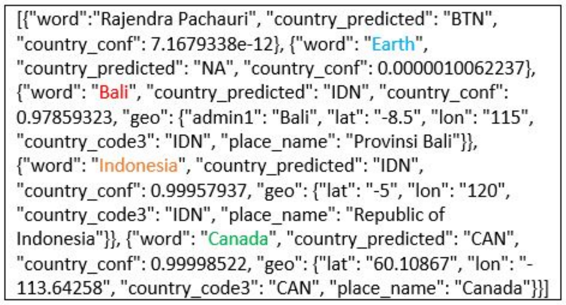
We discuss available geoparsers to process news and their drawbacks in Section II. Then we explain the proposed system architecture in detail in Section III. Next, we lay out the details of experiments conducted on the dataset using different geoparsers and the proposed system in Section IV. We end with concluding remarks and a discussion of future work.

II. Background

Various open source geoparsers were showcased and evaluated for performance in [3]. Apart from these, primary focus location extraction with 40% accuracy for security related events was featured in [5], and also Europe Media Monitor [6] is a fully automated system which parses news events worldwide for a few science areas of interest. Based on the performance and availability of the geoparsers, we briefly describe geoparsers like Profile [3], Cliff-Clavin [2] and Mordecai [4] in the following subsections.

TABLE II: Geoparsing 247.9K documents on a workstation with 10 cores

Geoparser	Total Processing Time	Speed
Cliff-Clavin	17.27 Mins	239.1 Docs/Sec
Profile	81 Mins	51.12 Docs/Sec
Mordecai	912 Mins	4.53 Docs/Sec



```
[{"word":"Rajendra Pachauri", "country_predicted": "BTN",
"country_conf": 7.1679338e-12}, {"word": "Earth",
"country_predicted": "NA", "country_conf": 0.0000010062237},
{"word": "Bali", "country_predicted": "IDN", "country_conf":
0.97859323, "geo": {"admin1": "Bali", "lat": "-8.5", "lon": "115",
"country_code3": "IDN", "place_name": "Provinsi Bali"}},
{"word": "Indonesia", "country_predicted": "IDN",
"country_conf": 0.99957937, "geo": {"lat": "-5", "lon": "120",
"country_code3": "IDN", "place_name": "Republic of
Indonesia"}}, {"word": "Canada", "country_predicted": "CAN",
"country_conf": 0.99998522, "geo": {"lat": "60.10867", "lon": "-
113.64258", "country_code3": "CAN", "place_name": "Canada"}}]
```

Fig. 3: Output of Mordecai for the sample news report

A. *Cliff-Clavin*

Cliff-Clavin [2] is an open source geoparser which is also hosted as a web service that parses news articles or other documents. It employs context-based geographic disambiguation over organizations and locations extracted from the text using Stanford CoreNLP. The focus places are identified from places mentioned at city, state, and country levels using a simple frequency-based method. Figure 2 illustrates the result on geoparsing sample news report from Figure 1 using Cliff-Clavin. It identified all locations mentioned in a document and also identified “Indonesia”, “Canada”, “Bali” as focus locations. We can observe from Table II that Cliff-Clavin geoparses unstructured text the fastest. Although it is fast in geoparsing text, it fails to identify a single focus location confidently. We can leverage the fact that the server can process hundreds of requests in parallel and can efficiently identify coordinates of all locations mentioned in a report.

B. *Mordecai*

Mordecai [4] is also an open source geoparser which uses MITIE’s NER tool to extract place names from text and then uses gazetteer in an elasticsearch index to identify focus country and all other place names from the text. It also fails to identify primary focus location of a report. A typical result of Mordecai is shown in Figure 3 which contains all mentioned location information from the sample news report in Figure 1. From Table II, we observe that Mordecai processes documents rather slowly compared to other geoparsers and thus it is not feasible to process millions of records due to the requirement of a large amount of resources to compensate for the low processing power of Mordecai.

C. *Profile*

Profile [3] identifies a primary focus location associated with a document using either MITIE or Stanford’s NER. For example, Figure 5 exhibits the results of Profile for the sample

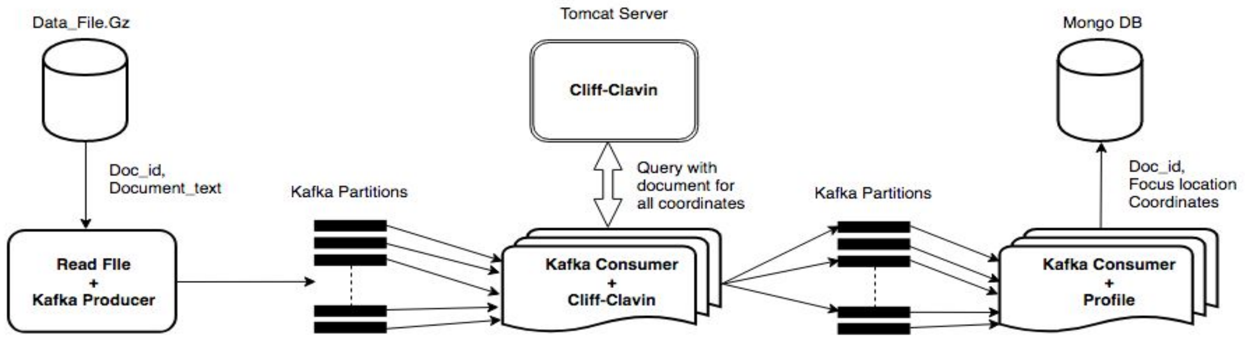

Data_File.Gz
Doc_id, Document_text
Read File + Kafka Producer
Kafka Partitions
Tomcat Server
Cliff-Clavin
Query with document for all coordinates
Kafka Consumer + Cliff-Clavin
Kafka Partitions
Mongo DB
Doc_id, Focus location Coordinates
Kafka Consumer + Profile

Fig. 4: System architecture of Sperg.

Fig. 5: Output of Profile for the sample news report

news report from Figure 1 and it can be seen that "Bali" was identified as a primary focus location. We can observe from Table II that Profile is slower than Cliff-Clavin but it geoparses ten times faster than Mordecai. Although Profile identifies the primary focus location, it fails to provide the necessary geo coordinates. Also, a single processor consumes about 6GB of main memory when running Profile which would prove to be costly when deployed in a multiprocessor environment.

All three geoparsers were found not to scale well when used to process millions of documents. Implementing the geoparsers in a multithreaded or multiprocessor environment is vital to process millions of documents in a short duration and to build a reliable fault-tolerant system. We propose a distributed framework named Sperg, which combines the results of Cliff-Clavin and Profile to extract coordinates of the primary focus location associated with a document.

III. System Overview

The architecture is laid out across multiple servers, and the message transfer between servers was carried out via Apache Kafka. The system operations can be broken down into three major groups of individual tasks which are:

- Read and distribute news reports data from source file;
- Use Cliff-Clavin to identify coordinates of all the locations mentioned in the report;
- Identify focus location using Profile and obtain coordinates by mapping with results obtained from Cliff-Clavin.

Each task depends on the output of the previous task and the whole pipeline can process documents in parallel as long as the rate of data flow is greater than what the pipeline can process within a certain period of time. Fig. 4 shows the system architecture of Sperg. The framework components and the optimizations performed over them are discussed further in the subsequent subsections in detail.

A. *Distributing Data Source*

We sourced our data from a single compressed file containing 76.1 million news reports totalling to 37GB. To geoparse all these documents efficiently, we relied on a parallel processing approach across multiple servers. We use message buffers such as Apache Kafka to queue up documents and consume them asynchronously from multiple servers. We chose Kafka for its highly scalable nature and fault tolerance. A simple python program read documents from the file and queued data in Kafka under a topic named cliff_input. A single threaded file reader was sufficient for the task since the major bottleneck in the system is geoparsing which is discussed in detail in the following subsections.

B. *Geo-coordinate Extraction*

We use a multithreaded program to parallelize the I/O intensive process across multiple sever. Each thread consumes documents from topics using Kafka consumer API. Each thread places an HTTP request to the Cliff-Clavin server with the consumed document, and receives a response from the server containing coordinates of all locations mentioned in the document and other information. We append the geographic information obtained from Cliff-Clavin onto the message with a unique delimiter and then publish the message onto a different Kafka topic named cliff_result. These threads continue the process as long as there are messages in the queue and until a certain timeout period.

C. *Focus Location Coordinate Mapping*

We make use of Profile to identify the primary focus location of a news report. In the next two subsections, we present major components of Profile with modifications to support multiple languages.

1) Word Embedding: Profile uses a feature extraction algorithm using a pre-trained word embedding model from raw text. It utilizes the publicly available fastText_multilingual [7] library which was built with fastText from Facebook and Google Translate API to align monolingual vectors from two languages in a single vector space. The length of these vectors is 300. It initialized the words that are not present in the set of pre-trained words to zeros. An interesting property of

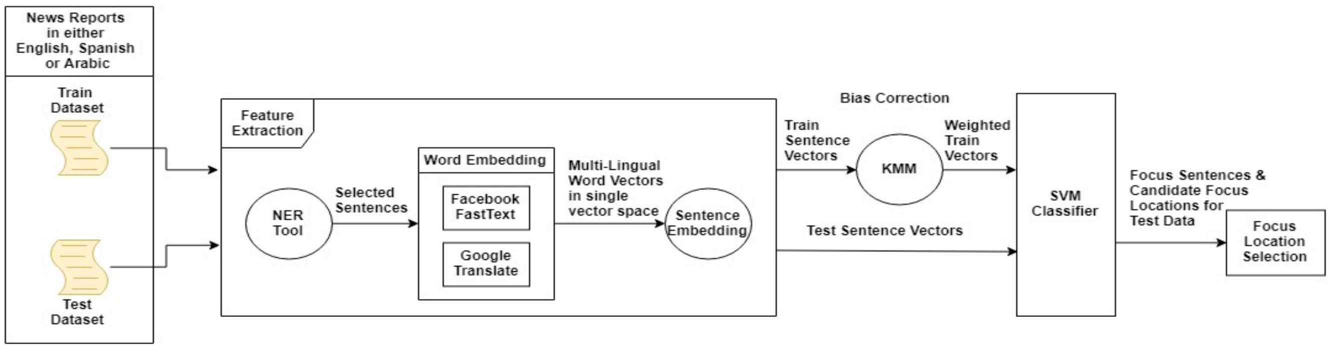


Fig. 6: A high level schema of modified Profile (Primary Focus Location Extraction).

word embedding is that these vectors effectively encode the semantic meanings of the words in the context. In other words, they are able to represent meaningful syntactic and semantic regularities in a very simple way [8].

2) Sentence Embedding: A basic sentence feature extraction method is also employed by Profile following the Sentence Embedding technique specified in [9]. In this approach, uncommon words are given more weight in the corpus, which essentially makes common words become less important in the dataset. An alternative approach to finding the sentence vector is by computing the mean of word vectors in the sentence. The effectiveness of this approach was empirically compared with that of another scheme of sentence embedding by assigning different weights to each word as shown in [3].

Lastly, Profile uses a learning method to identify the primary focus location from an unstructured text. Classification with limited training data [10] is a challenge and profile uses appropriate bias correction methods to overcome it. A high level schema of the modified Profile is shown in Figure 6. We obtain the primary focus location of a document using Profile which is a processer intensive module. So we utilize Profile from a multiprocessor environment with each processor serving a dedicated consumer to consume messages from cliff_result Kafka topic. Each message is parsed and we run the unstructured text through Profile to identify the primary focus location. We then look up the result from Cliff-Clavin for the coordinates of the identified primary focus location. The results from processing each message are written to a MongoDB instance. This setup and process is duplicated across multiple servers with coordination managed through Kafka.

D. Optimizations

We chose a batch size of 6 million documents to be sent to the Kafka topic so that there would be minimal reprocessing performed in case of a system failure. Moreover, we attained maximum parallelism when we equalled the number of partitions of the topic in Kafka to the total number of consumers across multiple servers which further avoided resource wait time since each consumer consumes from its own dedicated partition. We used one dedicated server with additional disks to host Kafka and MongoDB in order to balance load on memory and disk consumed by the pipeline since other processes of the pipeline are memory intensive. To improve MongoDB write performances, we disabled journaling and also created an index on certain keys after processing all the documents. Indexing was required to query faster and was performed later because insertion tends to be slower for an indexed collection as the indices have to be re-calibrated after every insertion.

TABLE III: Datasets Statistics

Dataset	Documents	Size
Terrier Event Data	76.1M	76GB
Randomly Sampled Terrier Event Data	247.8K	290MB
Atrocity Event Data Training data for profile	3.6K	9.3KB

The word embedding approach used by Profile involves loading Facebook's fasttext vectors (at most 6GB vector file) into memory which leads to higher resource demand when implemented in a multiprocessor environment capable of processing millions of reports. We leveraged the fact that fasttext vectors were required only in the read operations, and with the help of the basemanager module available in the multiprocessing python package, we hosted fasttext vectors as a service which could be shared across multiple processors for read operations. Therefore, the total memory used by Profile in a multiprocessor environment is reduced significantly by almost 80%.

IV. Experiments

A. Dataset

Table III portrays the size of the datasets and number of documents used to experiment on the proposed framework. The Atrocities Event Data [11] is a collection of English news reports on atrocities and mass killings in several locations. Human coders have read the reports and extracted metadata about the events. The annotated reports include focus locations, and the reports that captured the event. This dataset was used as training data for Profile. The Terrier [12] (Temporally Extended Regularly Reproducible International Event Records) data contained encoded event data for 76 million news reports which was prepared with assistance from [13]. We randomly sampled 250K documents to tune

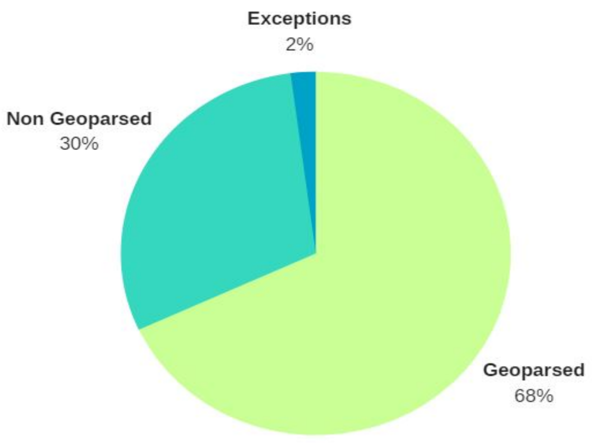


Fig. 7: Distribution of 76.1 million documents after Sperg processing

TABLE IV: Distribution of 51.83 million documents geoparsed by Sperg. ✓ and ✗ indicates whether the corresponding geoparser found the location or not.

Case	Cliff Clavin Result	Profile Result	Documents count (in Millions)
I	✓	✓	47.74
II	✓	✗	4.09

TABLE V: Distribution of 22.68 million documents non geoparsed by Sperg. ✓ and ✗ indicates whether the corresponding geoparser found the location or not.

Case	Cliff Clavin Result	Profile Result	Documents count (in Millions)	Possibility to extract coordinates ?
I	✓	✓	11.26	Yes
II	✗	✓	2.81	Yes
III	✓	✗	2.87	Maybe
IV	✗	✗	5.74	No

the performance of our pipeline before we run the data set through Sperg.

B. Environment Specifics

We used XSEDE resources [14], two s1 xxlarge machines with 44 cores and 120 GB memory. Using this resource, our system was deployed on 6 VMs each with an Intel® E5-2680v3 Haswell CPU @ 2.50GHz, 10 cores, 29 GBs of RAM, 239 GBs of storage and an additional 2TB disk attached to one of the node. All the processing in these experiments was carried out with 50 total cores and 145 GBs of RAM. Ten total cores and 29 GBs of RAM on a single node were allocated for storage and message buffers in these experiments. All of the six nodes were located in Indiana under the same network.

C. Application Configurations

We hosted Apache Kafka and MongoDB on a single node with the 2TB disk for seamless service. We created two topics namely cliff_input with 25 partitions and cliff_results with 50 partitions on Kafka. These topics shared similar configurations such as log.retention time of at least two days, log.segment size of 100Mb and log.retention.check.interval of 30 seconds for efficiency. In the same server as Kafka and MongoDB, a python script was used to read six million documents from the source file at a fixed interval and publish the data into Kafka cliff_input topic. Instances of Profile and Cliff-Clavin applications were deployed on the remaining five nodes. A python script with five threads consumed documents from Kafka cliff_input topic, sent post requests to Cliff-Clavin server with the text for the coordinates and then publishes the coordinates along with the text to Kafka cliff_result topic asynchronously.

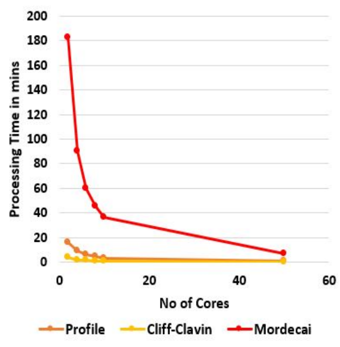
To achieve parallelism, Profile is hosted in a multiprocessor environment with each processor responsible for 1) Consuming text and coordinates from Kafka cliff_result topic, 2) Identifying the primary focus location from the given text, 3) Matching the primary focus location with the result from Cliff-Clavin for the coordinates, and 4) Writing mapped results onto a MongoDB collection named as geoloc with the coordinates and document_id. The nonmatching results are stored in a different collection named as missmatchloc with document_id and results of Profile and Cliff-Clavin.

D. Evaluation

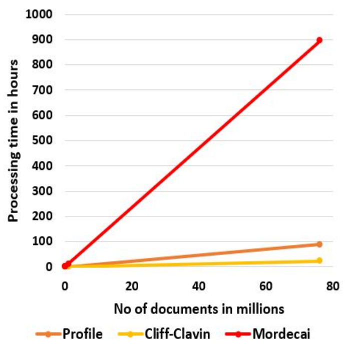
Sperg processed 76.1 million documents in 5.29 days inclusive of two server failures that demanded reprocessing of two batches. Fig. 7 depicts that 68% of the total documents were geoparsed successfully and 30% are not geoparsed, i.e. there was a mismatch between the results of Profile and Cliff-Clavin and 2% yielded exceptions through various stages of Sperg. Table IV is a report on the geoparsed documents, Case I is when results of Cliff-Clavin and Profile match and Case II is when Profile fails to identify the primary focus location while Cliff-Clavin gave a single location which is taken as the primary focus location for the document by Sperg.

Sperg failed to identify the primary focus location coordinates for 22.68 million documents and the potential for these documents to be geoparsed to fetch a primary focus location coordinate is recorded in Table V. Profile identified the primary focus location for case I and II but could not match the results of Cliff-Clavin, so there is a chance to extract coordinates using other geoparsers and match the result of Profile with it. Case III is when Profile did not give any result but Cliff-Clavin identified multiple locations within the document, from which the coordinates of primary focus location could be extracted with a frequency-based algorithm. Case IV is when both Cliff-Clavin and Profile failed to give results and since both used different NERs, it is more probable that these documents do not have a location mentioned in them.

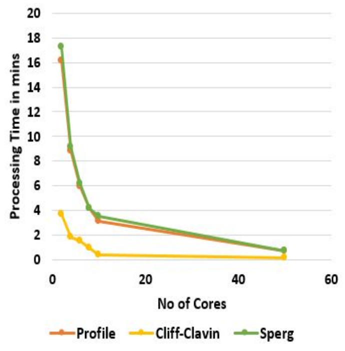
Fig. 8 provides a visual on the performance of geoparsers and Sperg. Fig. 8a and 8b compares the performance of Cliff-Clavin, Profile and Mordecai on 10K documents with multiple cores and processing up to 76.1 million documents on 50 cores respectively. Fig. 8a proves the fact that processing time



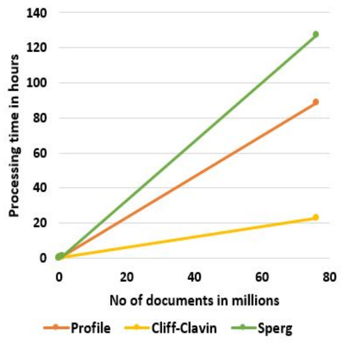
(a) Performance on 10K documents



(b) Performance on 50 cores



(c) Performance on 10K documents



(d) Performance on 50 cores

Fig. 8: Comparison of Cliff-Clavin, Mordecai, Profile and Spergs performance.

reduces almost linearly upon adding more cores, i.e. geoparsing across more servers. Although we reduce processing time by increasing the number of cores, it could prove to be costly in the case of Mordecai because it requires at least ten times more resources compared to Profile and Cliff-Clavin. We also observed from Fig. 8b that Mordecai, Profile and Cliff-Clavin would take 37.3, 3.6 & 0.9 days respectively to geoparse 76.1 million documents. These results prove that use of Mordecai is not feasible to process documents given the limited number of resources and timeframe. Performance of Sperg is more dependent on performance of Profile as seen from Fig. 8c and 8d, since Cliff-Clavin can parse 76.1 million documents within 24 hours while Profile would complete the same task in almost 4 days. Therefore, Sperg is configured to produce throughput equal to that of Profile, but observed throughput is lower compared to Profile in Fig. 8c and 8d because of network latency and system failures.

V. Conclusion and Future Work

In this paper we described a scalable distributed framework, Sperg, for extracting geolocation coordinates for the events from political news reports. Cliff-Clavin and Profile were identified as ideal candidates to constitute Sperg in solving the problem. Sperg successfully geoparsed 51.83 million documents and is much more flexible due to the capability to add or remove servers dynamically as per the requirement or the problem size. This architecture provides a platform for researchers to geoparse big data and due to its dynamic nature, it provides confidence in building a real time system.

Our future work includes: (i) Support for multilingual report geoparsing using the capabilities of Profile, (ii) Build a real time system to process live new reports on the go, (iii) Add more geoparsers to our federated system to process the remainder of the documents which Sperg failed to geoparse, (iv) Configure Sperg to support high availability, i.e. process documents seamlessly even in the event of a system failure.

Acknowledgment

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562. This material is based upon work supported by National Science Foundation (NSF) award number SBE-SMA-1539302.

References

[1] V. A. Petrushin and L. Khan, *Multimedia Data Mining and Knowledge Discovery*. Springer-Verlag London, 2007, ch. Multimedia Data Mining: An Overview.

[2] C. D'Ignazio, R. Bhargava, E. Zuckerman, and L. Beck, "Cliff-Clavin: Determining geographic focus for news," *NewsKDD: Data Science for News Publishing, at KDD 2014*, 2014.

[3] M. B. Imani, S. Chandra, S. Ma, L. Khan, and B. Thuraisingham, "Focus location extraction from political news reports with bias correction," in *Big Data (Big Data), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1956–1964.

[4] mordecai, "[online]," *URL: Available: https://github.com/openeventdata/mordecai*.

[5] M. Atkinson, J. Piskorski, H. Tanev, and V. Zavarella, "On the creation of a security-related event corpus," in *Proceedings of the Events and Stories in the News Workshop*. Association for Computational Linguistics, 2017, pp. 59–65. [Online]. Available: http://aclweb.org/anthology/W17-2709

[6] E. M. Monitor, "[online]," *URL: Available: http://emm.newsbrief.eu/*.

[7] S. H. Samuel L. Smith, David H. P. Turban and N. Y. Hammerla, "Offline bilingual word vectors, orthogonal transformations and the inverted softmax," *arXiv preprint arXiv:1702.03859*, 2017.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[9] S. Arora, Y. Liang, and T. Ma, "A simple but tough-to-beat baseline for sentence embeddings," in *International Conference on Learning Representations. To Appear*, 2017.

[10] M. M. Masud, J. Gao, L. Khan, J. Han, and B. Thuraisingham, "Classification and novel class detection in data streams with active mining," in *Advances in Knowledge Discovery and Data Mining*, M. J. Zaki, J. X. Yu, B. Ravindran, and V. Pudi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 311–324.

[11] P. Schrodt and J. Ulfelder, "Political instability task force worldwide atrocities dataset," *Lawrence, KS: Univ. Kansas, updated*, vol. 8, 2009. [Online]. Available: http://eventdata.parusanalytics.com/data.dir/atrocities.html/

[12] A. Halterman, J. Irvine, M. Landis, P. Jalla, Y. Liang, C. Grant, and M. Solaimani, "Adaptive scalable pipelines for political event data generation," in *2017 IEEE International Conference on Big Data (Big Data)*, Dec 2017, pp. 2879–2883.

[13] O. Tange, *GNU Parallel 2018*. Ole Tange, Mar. 2018. [Online]. Available: https://doi.org/10.5281/zenodo.1146014

[14] J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G. D. Peterson, R. Roskies, J. R. Scott, and N. Wilkins-Diehr, "Xsede: Accelerating scientific discovery," *Computing in Science Engineering*, vol. 16, no. 5, pp. 62–74, Sept.-Oct. 2014. [Online]. Available: doi.ieeecomputersociety.org/10.1109/MCSE.2014.80