# Big Data Analytics for Concurrent Data Processing

A Samydurai
Department of Computer Science and Engineering
Valliamai Engineering College
SRM Nagar, Kattankulathur, India

C Vijayakumaran
Department of Computer Science and Engineering
Valliamai Engineering College
SRM Nagar, Kattankulathur, India

G Kumaresan
Department of Computer Science and Engineering
Valliamai Engineering College
SRM Nagar, Kattankulathur, India

B Muthusenthil
Department of Computer Science and Engineering
Valliamai Engineering College
SRM Nagar, Kattankulathur, India

## ABSTRACT
Attractively voluminous data describes an immense volume of structured and unstructured data that is difficult to process utilizing traditional database techniques. The tremendous growth in arrival rates of data to support a large number of user queries creates complex problems in the traditional structured databases. In this paper, the input file is assigned to a master who has the ability to split and control the work flow with different workers. This will reduce the fault tolerance issues raised with nodes. They will evaluate the intermediate files and data items. Over again the processed data will be amalgamated and the required output will be immediately/middle file given to the user. Also the first solution for processing perpetual text queries efficiently to address the above challenges is given. The solution indexes the streamed documents in main recollection with a structure predicate on the principles of the inverted file, and processes document advent and expiration events with an incremental threshold-predicated method.

## Keywords
Structured and unstructured data, Fault Tolerance, Perpetual Text Queries, Structure Predicate, Inverted File and Threshold-Predicated Method.

## 1. INTRODUCTION
The capacity of information in our world has been rising explosively, and analyzing huge information sets therefore referred to as "Big Data". It becomes a key basis of competition reinforcement, new waves of productivity growth, novelty, and surplus customer support. Massive information refers to information sets whose size is on the far side the power of current technology, technique and theory to capture, manage, and method the information inside a tolerable time period. Today, massive information management stands out as a challenge for IT firms. The answer to such a challenge is shifting progressively from providing hardware to provide more manageable software package solutions massive information conjointly brings new opportunities and significant challenges to business and academe. With the growing variety of other services, effectively recommending services that users most popular has become a vital analysis issue. Service recommender systems are shown as valuable tools to assist users handle services overload and supply acceptable recommendations to them [17]. The model of pool management is enforced to produce effective resource management and a cache memory is provided to diminish redundancy occurring because of same user requests. The gains obtained are a higher utilization of resources and reduced time interval from source to destination [18].

The exaggerated use of digital information channels, like email, electronic news feeds, and automation of business news functions, as well as the importance of creating timely choices, raise the requirement for a nonstop text search model. During this model, new documents hit an observation server within the style of a stream. The server hosts several text search queries that square measure put in once and stay active till terminated by the users. Every question letter unceasingly retrieves, from a window of the foremost recent documents, that's most kind of like a set of search terms.

For instance, a security analyst who monitors email traffic for potential terror threats would register many standing queries to spot recent emails that the majority closely match sure risk profiles. A word associated with the industries of interest is developed as standing text queries over the newsflashes.

Previous studies on document filtering have targeted on techniques for adaptively setting a similarity threshold to see whether or not every document has relevancy to a question. However, the matter of with efficiency maintaining the list of the k most relevant documents has not been thought of. Existing schemes for continuous top-k process, on the opposite hand, deem special index structures. In text retrieval, every term within the lexicon is taken into account a dimension. At its core lies a memory-based index like the traditional inverted file, complemented with quick update techniques and book-keeping structures. We have a tendency to cipher the first-time results of a question with a threshold-based rule on the inverted lists. The thresholds derived square measure used for sequent result maintenance, and specifically one. The paper organized related works in section 2, problem statement in section 3, general design for the big data architecture in section 4, performance analysis with GUI model in section 5 and finally conclusion and future work of the paper.

## 2. RELATED WORK
The author discussed, we have a tendency to plan a replacement reshuffling strategy in Hadoop to cut back high network masses obligatory by shuffle-intensive applications. Coming up with new shuffling methods is extremely appealing for Hadoop clusters wherever network interconnects square measure performance bottleneck once the clusters square measure shared among an outsized range of applications. The network interconnects square measure possible to become scarce resource once several shuffle-intensive applications square measure sharing a Hadoop

cluster [1]. The author planned, presents design to face this issue. The planned resolution relies on the combination of rule-based system with sensor-cloud computing surroundings [2]. The author identified a number of the perquisites for reconfigurable computing systems within the cloud and picks out many eventualities created attainable with cloud-based computing capability [3].The author putforth, we tend to gift AROM, a framework for big scale distributed process supported DFG to precise the roles and that uses paradigms from practical programming to outline the operators. the previous results in a lot of natural handling of pipelined tasks whereas the latter enhances generosity and reusability of the operators, as exposed by our tests on a equivalent and pipelined job activity the calculation of Page rank [4]. We introduce mechanisms that change caching in such networks, whereas maintaining the most principle of loose coupled and asynchronous communication. moreover we have a tendency to investigate 2 caching policies; caching altogether candidate brokers (basic caching) that yields high survivability and low delay and caching in leaf brokers (leaf caching) that maintains low overhead and querying complexness. The comparison is performed via simulations and work measurements and insights are given for future work [5]. DITTO exploits properties specific to invariant checks to modify and modify the method while not constrictive the varieties of mutations which will be performed. The source-to-source implementation of DITTO, for Java, is automatic, portable, and economical, providing speedups on knowledge structures with as few as a hundred components urged the author [6]. In this summary, we have a tendency to encourage the requirement for and analysis problems arising from a replacement model of knowledge process. during this model, knowledge doesn't take the shape of persistent relations, however rather arrives in multiple, continuous, rapid, time-varying knowledge streams mentioned the author [7].The author projected, we tend to introduce the key techniques within the space, describing each a core implementation and the way the core is increased through a spread of extensions. we tend to conclude with a comprehensive list of text compartmentalization literature [8].This presents a brand new technique of adjusting dissemination thresholds that expressly models and compensates for this bias. The new algorithmic program, which relies on the most chance principle, conjointly estimates the parameters of the density distributions for relevant and non-relevant documents and therefore the quantitative relation of the relevant document within the corpus. Experiments with TREC-8 and TREC-9 Filtering Track knowledge demonstrate the effectiveness of the algorithmic program projected by the authors [9].We get two process techniques: the primary one computes the new answer of a question whenever a number of this top-k points expire; the second partly pre-computes the longer term changes within the result, achieving higher period at the expense of slightly higher area necessities. we tend to analyze the performance of each algorithms and judge their potency through intensive experiments. Finally, we tend to extend the projected framework to different question varieties and a unique knowledge stream model the author projected [10].The author planned associate analysis technique that uses early recognition of that documents area unit possible to be extremely stratified to cut back costs; for our take a look at knowledge, queries area unit evaluated in two of the memory of the quality implementation while not degradation in retrieval effectiveness. C.P.U. time and disk traffic can even be dramatically reduced by coming up with inverted indexes expressly to support the technique [11]. The author represented, a replacement inverted file structure mistreatment quantity weights that has superior retrieval effectiveness compared to standard inverted file structures once early termination heuristics area unit utilized. That is, we tend to area unit able to reach similar effectiveness levels with less procedure value, so offer a more robust cost/performance compromise than previous inverted file organizations [12]. The authors given results based mostly upon the trec net knowledge that show the mixture of those varied techniques to yield extremely competitive retrieval, in terms of each effectiveness and potency, for each short and long queries [13]. The author delineated optimization techniques that may scale back question analysis prices. Presents simulation results that compare the performance of those optimization techniques once applied to tongue question analysis (JMV)[14].T he author mentioned, we tend to explore different question analysis techniques, and develop new techniques for evaluating queries on passages. we tend to show by experimentation that, suitably enforced, effective passage retrieval is sensible in restricted memory on a desktop machine[15].We simulated our technique to explore the matter area, then enforced it in indri, our giant scale language modeling programme. Tests with the GOV2 corpus on title queries show our technique to be twenty third quicker than max_score alone, and sixty one quicker than our document-at-a-time baseline. Our optimized question times square measure competitive with typical term-at-a-time systems on this year's TREC computer memory unit task is author planned [16].

## 3. PROBLEM STATEMENT

Consider a text filtering server that monitors a stream of incoming documents for a collection of users, their interests within the style of continuous text search queries. The task of the server is to perpetually maintain for every question a graded result list, comprising the recent documents with the very best similarity to the question. Such a system underlies several text observation applications that require addressing significant document traffic, like email and news observation. The disadvantage is that the we tend to might realize heap of duplication files that cause high search time, no run time updation of in sequence, query is graded supported the recent documents. The probable system is that the 1st resolution for process continuous text queries expeditiously. Our objective is to support an oversized variety of user queries whereas sustaining high document arrival rates. Our resolution indexes the streamed documents in main memory with a structure supported the principles of the inverted file; processes file appearance and running out events with a progressive threshold-based technique. In this paper, we tend to propose the theme of window that updates the newest info with reference to time and updated knowledge. We tend to additionally propose to get rid of the duplication of the Records within the on-line. We learn to furthermore get the reaction from the previous users of the consequent on-line site. Therefore as a final point we learn to put into operation the ranking method by obtaining the feedback from the users, removal of duplication together with window approach. The advantage is rule is employed to eliminate the duplication so the load of the server is reduced. Change the file at runtime is displayed once the user searches for a question. Algorithmic rule is employed to chop the unwanted words from files.

## 4. GENERAL DESIGN

A system benchmark would embrace the entire channel therefore additionally referred to as an end-to-end benchmark.

An element benchmark would live and take a look at solely a part of the system. Several members of the massive information community want a benchmark for the whole system, but it's open question whether or not such a benchmark will be created that's helpful and sensible. The model gets raw data from outside file, and split the information as messages and assign to workers. Then workers process the requested queries from user through a master who has the ability to split and control the work flow with different workers as shown in Figure 1.

The diagram shows the overview of the usage requirement of the proposed system. They are useful for presentation to stakeholders, but for actual development we will need to find that use cases provide significantly more value because they describe "the meant" of the actual requirements. A diagram describes a sequence of action that provides something of measurable value to an action and is drawn as a horizontal ellipse as shown in Figure 2. The modules can be classified as the user, data server, sliding window, duplication detection and user feedback. The user module searches for the information. For the ordinary search, lot of results will be displayed with the duplications and the random display of outputs. In the proposed work, the user can give the query and accordingly the updated information is provided. The data server will display the details with all possible output with repetitive information and updates at the end in proper order. The data server is used to retrieve the exact updated information with no duplicates by the use of sliding window technique and with the help of Unsupervised Duplication Detection (UDD). Hence data server performs the above operation and gives the result for the given query. The sliding window is the sliding window is the technique which utilizes both time and data updation. In case of time, if the information provided in the site is updated later then the last updated data will be displayed first. In case of data, if the old data is updated then the updated information will be displayed according to the priority. The duplication detection is the paper, to avoid the repetition of same data result we utilize Unsupervised Duplication Detection algorithm. By using this algorithm, we can avoid the duplicate data and the results will be displayed to the user without any duplicates.

The user feedback module will ask user to provide the feedback regarding the searched site. Based upon the feedback, ranking will be provided for the site. If the user provides positive feedback then the ranking will increment the page ranking and if the feedback is negative, it will be decremented. Accordingly the best ranking results will be displayed in that refined search site. Finally the data retrieval is doing for the search by means of these techniques; and it will be efficiently done. Hence, data duplication is avoided and also the latest innovated information is provided to the user.

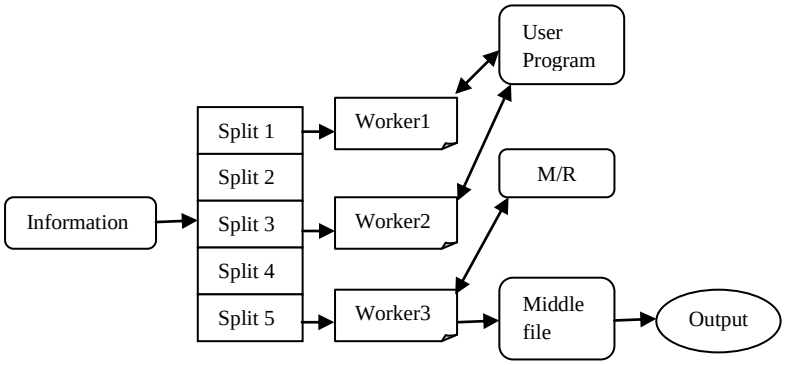


**Fig 1: A Model of the Big Data Design**

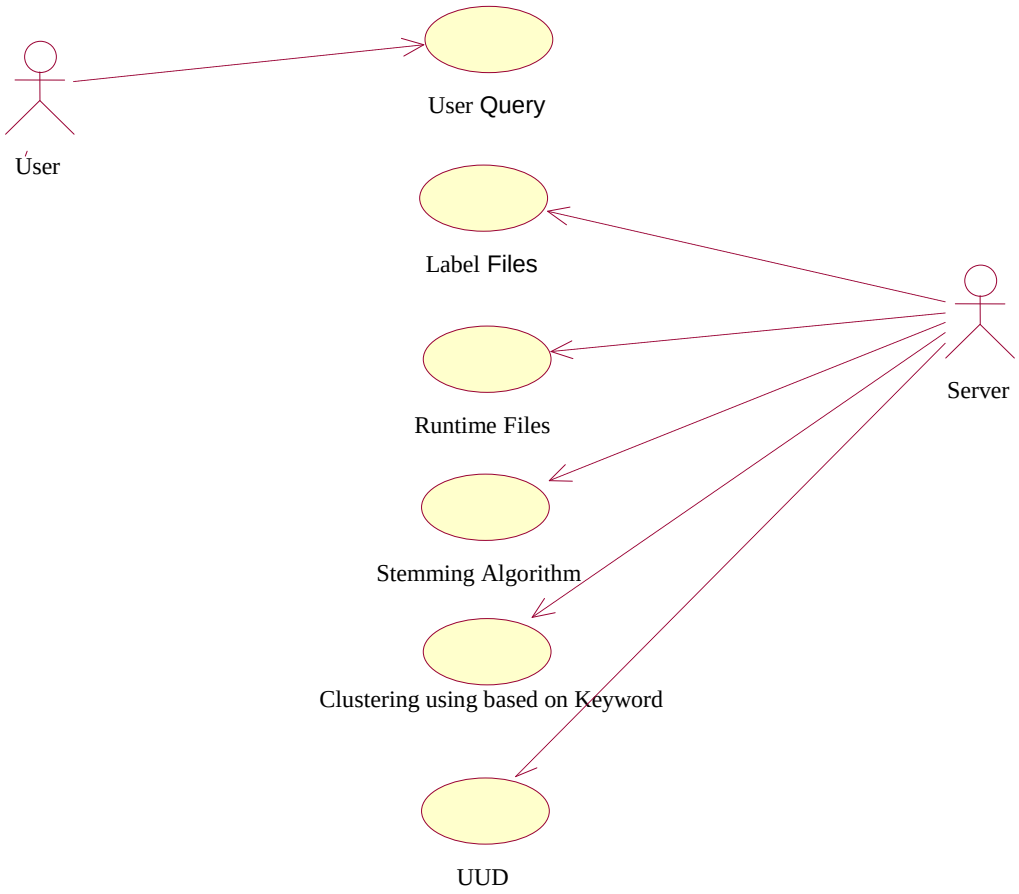


**Fig 2: Use Case diagram for data processing**

## 5. PERFORMANCE ANALYSIS

The sample GUI screen of the proposed method is shown in Figure 3. Here, the user interface is designed in Java with MySQL Server. Each client connection gets its own thread within the server process. When clients (applications) connect to the MySQL server, the server needs to authenticate them. Before even parsing the query, though, the server consults the query cache, which only stores SELECT statements, along with their result sets. The storage engine affects how the server optimizes query. MySQL supports several storage engines that act as handlers for different table types. MySQL storage engines include both those that handle transaction-safe tables and those that handle non-transaction-safe tables.

For datasets that fit in memory, the proposed method offers substantial performance gains over file-based map/reduce platforms. To measure the effect of these optimizations, performance measurements were made for the stock trading analysis described above using Scale-out Analytics Server as the number of stock histories and grid servers was proportionally increased. As shown in the Table 1, various throughput comparisons with Hadoop. Initially during one server execution, the throughput for PMI is 100, Scale out is 300 and Hadoop is 1000, as per our data it shows more throughput data transfer through the Hadoop compare to others. Similarly, whenever number of server increases, the data transfer rate is also an increase which in turn increases the throughput in Hadoop.

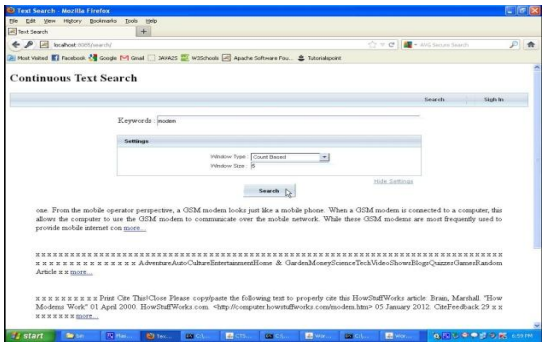


**Fig 3 The Scheme for text retrieval**

As the graph below illustrates, the UDD delivered linearly scalable throughput (shown as the red line in the graph). An alternative implementation of this application was measured using Hadoop's map/reduce environment. Hadoop provided linear scaling with about 16 times lower throughput (shown as the blue line in the graph shown in Figure 4) due to significant overhead introduced by file I/O, combining, reducing, and batch scheduling. To see the impact of file I/O, data was staged in the UDD instead of the Hadoop file system (HDFS). Hadoop's throughput was increased by about 6 times (shown as the green line). The remaining difference is largely due to file I/O between the map and reduce phases. Although Hadoop is working to reduce this additional file I/O, significant data motion and lower performance due to the use of multiple reducers is unavoidable.

**Table 1 Various Throughput comparisons with Hadoop**

| Number of server | Scale Out PMI | Hadoop/ Scale Out | Hadoop |
|---|---|---|---|
| **1** | 100 | 300 | 1000 |
| **2** | 200 | 600 | 2000 |
| **3** | 300 | 900 | 3000 |
| **4** | 400 | 1200 | 4000 |

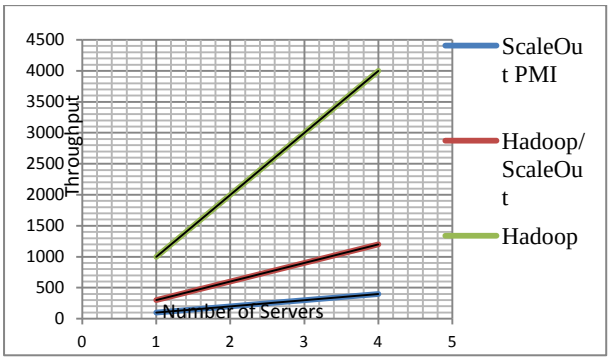


**Fig 4 Performance of Hadoop on Throughput**

## 6. CONCLUSION

For datasets that fit in memory, the proposed method offers substantial performance gains over file-based map/reduce platforms. To measure the effect of these optimizations, performance measurements were made for the stock trading analysis described above using Scale-out Analytics Server as the number of stock histories and grid servers was proportionally increased. As the graph below illustrates, the UDD delivered linearly scalable throughput (shown as the red line in the graph). An alternative implementation of this application was measured using Hadoop's map/reduce environment. Hadoop provided linear scaling with about 16 times lower throughput (shown as the blue line in the graph shown in Figure 4) due to significant overhead introduced by file I/O, combining, reducing, and batch scheduling. To see the impact of file I/O, data was staged in the UDD instead of the Hadoop file system (HDFS). Hadoop's throughput was increased by about 6 times (shown as the green line). The remaining difference is largely due to file I/O between the map and reduce phases. Although Hadoop is working to reduce this additional file I/O, significant data motion and lower performance due to the use of multiple reducers is unavoidable. In future, big data platforms are replacing traditional data infrastructure, providing capability and performance will increase in progressive costs, compared with traditional infrastructure exponential costs. Hadoop made its debut as the future of big data, with cheaper data storage and faster processing.

## 7. REFERENCES

[1] Jiong Xie, FanJun Meng, HaiLong Wang, JinHong Cheng, Hongfang Pan and Xiao Qin, "Adaptive Preshuffling In Hadoop Clusters", International Journal of Grid and Distributed Computing, Vol. 6,, No. 2, April 2013, pp. 79-92

[2] Saymon Castro de Souza, José Gonçalves Pereira Filho, and Eugênio Fraga Spessimille, "A Rule-Base Approach for WSN Application Development in a Cloud Environment", JACN 2013 Vol.1(4): 306-309 ISSN: 1793-8244.

[3] Anil Madhavapeddy, Satnam Singh, "Reconfigurable Data Processing for Clouds", IEEE 19th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), 2011, pp. 141 - 145

[4] Nam-Luc Tran and Sabri Skhiri, Arthur Lesuisse and Esteban Zim´anyi, "AROM: Processing Big Data With Data Flow Graphs and Functional Programming", IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom), 2012, pp. 875 – 882

[5] Vasilis Sourlas, Georgios S. Paschosy, Paris Flegkas_ and Leandros Tassiulas, "Caching in content-based publish/subscribe systems", IEEE International Conference on Global Telecommunications Conference, 2009. GLOBECOM 2009, pp. 1 – 6.

[6] Ajeet Shankar, Rastislay Bodik, " DITTO: Automatic Incrementalization of Data Structure Invariant Checks (in Java)", Proceedings of the 2007 ACM SIGPLAN conference on Programming language design and implementation, Volume 42 Issue 6, June 2007, Pages 310-319

[7] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and Issues in Data Stream Systems," Proc. Twenty-First ACM SIGMOD-SIGACT-SIGART Symp. Principles of Database Systems (PODS '02), pp. 1-16, 2002.

[8] J. Zobel and A. Moffat, "Inverted Files for Text Search Engines," ACM Computing Surveys, vol. 38, no. 2, pp. 1-55, July 2006.

[9] Y. Zhang and J. Callan, "Maximum Likelihood Estimation for Filtering Thresholds," Proc. 24th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '01), pp. 294-302, 2001.

[10] K. Mouratidis, S. Bakiras, and D. Papadias, "Continuous Monitoring of Top-k Queries over Sliding Windows," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '06), pp. 635- 646, 2006.

[11] M. Persin, J. Zobel, and R. Sacks-Davis, "Filtered Document Retrieval with Frequency-Sorted Indexes," J. Am. Soc. for Information Science, vol. 47, no. 10, pp. 749-764, 1996.

[12] V.N. Anh, O. de Kretser, and A. Moffat, "Vector-Space Ranking with Effective Early Termination," Proc. 24th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '02), pp. 35-42, 2001.

[13] V.N. Anh and A. Moffat, "Impact Transformation: Effective and Efficient Web Retrieval," Proc. 25th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '02), pp. 3-10, 2002.

[14] H.R. Turtle and J. Flood, "Query Evaluation: Strategies and Optimizations," Information Processing Management, vol. 31, no. 6, pp. 831-850, 1995.

[15] M. Kaszkiel, J. Zobel, and R. Sacks-Davis, "Efficient Passage Ranking for Document Databases," ACM Trans. Information Systems, vol. 17, no. 4, pp. 406-439, 1999.

[16] T. Strohman, H. Turtle, and W.B. Croft, "Optimization Strategies for Complex Queries," Proc. Research and

Development in Information Retrieval (SIGIR '05), pp. 219-225, 2005.

[17] V Shunmei Meng, Wanchun Dou, Xuyun Zhang, and Jinjun Chen,' KASR:AKeyword-AwareService Recommendation Method on Map Reduce for Big Data Applications', IEEE Transactions on Parallel and Distributed Systems, Vol. 25, No. 12, 2014, PP. 3221-3231.

[18] Samydurai, A , Amitha, T , Bhagyalakshmi.S, Aparna.G, Govardhani.S, "Effective and Efficient Utilization of Green Computing Based Optimized Resource in Cloud with VMware", International Journal of Applied Engineering Research, Vol. 10, No. 17, 2015, pp. 13499-13502.