Big data Analysis for frame work Collections

S.Nandhini devi[1], S.Nixan[2], Nithish Kumar[3]

MAMSCHOOL OF ENGINEERING, ANNA UNIVERSITY

[1]devinandhini1982@gmail.com

[2]Nixa25inno@gmail.com

[3]Nithish09091998@gmail.com

Abstract—Most scientific data consists of analyzing huge amount of data collected from different resources. Hence, parallel algorithms and frameworks are the important which can process huge volumes of data and meeting the requirements of performance and scalability entailed in such scientific data analyses. In this paper, we proposed a concurrent VM reconfiguration mechanism for big data tool which is Map Reduce on virtualized big data environments. Our reconfiguration enhances the input data locality of a virtual Map Reduce cluster. It adds cores to VMs to run local tasks temporarily by scheduling tasks based on data locality, and adjust the computational capability of the virtual nodes to contain the scheduled tasks unlike the traditional schemes which can leads to user-friendly configuration methods for big data resources. The need for a reliable mapping mechanism decreases these risks to a minimum. In distributed systems, faults or failures are limited or part. Mapping is a significant issue in big data; it is concerned with all the techniques necessary to enable a system to endure software faults remaining in the system after its development. The main benefits of implementing mapping in big data include failure recovery, lower cost, improved performance etc. When multiple instances of an application are running on numerous machines and one of the servers goes down, there exists a fault and it is implemented by fault tolerance. So in this project we can mapping mechanisms using failure detectors based on check points. In real time distributed system feasibility of task is very important because there is a deadline defined for each task and should be ended on or before its deadline even there is a fault in the system. This Paper aims to provide a better accepting of fault, mapping and mapping techniques used in the distributed real time environments.

Keywords—Mapping, Mining, Attribute Selection, fault tolerance.

I. INTRODUCTION

Distributed Computing Systems consists of variety of hardware and software components. Failure of any of these components can lead to unanticipated,potentially disruptive behavior and to service availability. Fault – Can be termed as "defect" at the lowest level of abstraction. It can lead to erroneous system state. Faults. may be classified as transient, intermittent or permanent. They can be of following g types: 1. Processor Faults (Node Faults): Processor faults occur when the processor behaves in an unexpected manner. It may be of classified into three kinds: a) Fail-Stop – Here a processor can both be active and participate in distribute protocols or is totally failed and will never respond. In this case the neighboring processors can detect the failed processor. b) Slowdown – Here a processor might run in degraded fashion or might totally fail. c) Byzantine – Here a processor can fail, run in degraded fashion for some time or execute at normal speed but tries to fail the computation. 2. Network Faults (Link Faults): Network faults occur when (live and working) processors are prevented from communicating with each other. Link faults can cause new kinds of problems like: a) One way Links – Here one processor can send messages to other is not able to receive messages. This kind of problem is similar to that faced due to processor slowdown. b) Network Partition – Here a portion of network is completely isolated with the other. Error – Undesirable system state that may lead to failure of the system

2. LITERATURE SURVEY

2.1 Mapping in Real Time Distributed System

A faulty system due to any reason during processing some task can causes some damages. A task running on real time distributed system should be feasible, reliable and scalable. The real time distributed system such as nuclear systems, robotics, air traffic control systems, grid etc. are highly dependable on deadline. A fault in real time distributed system can result a system into failure if not properly detected and recovered at time. These systems must function with high availability even under hardware and software faults. Fault-tolerance is the important technique used to maintain dependability in these systems. Hardware and software redundancy are well-known effective

2.1 Mapping in Real Time Distributed System

A faulty system due to any reason during processing some task can causes some damages. A task running on real time distributed system should be feasible, reliable and scalable. The real time distributed system such as nuclear systems, robotics, air traffic control systems, grid etc. are highly dependable on deadline. A fault in real time distributed system can result a system into failure if not properly detected and recovered at time. These systems must function with high availability even under hardware and software faults. Fault-tolerance is the important technique used to maintain dependability in these systems. Hardware and software redundancy are well-known effective methods. Hardware fault-tolerance achieved through applying extra hardware like processors, communication links, resource (memory, I/O device) whereas in software mapping tasks, messages are added into the system to deal with faults. Fault should be detected by applying reliable fault detector followed by some recovery technique. Many fault detection t techniques are available but it is necessary to apply appropriate fault detector. Unreliable fault detector can make mistake by erroneously suspecting correct process or trusting crashed process. Main focus is on hardware mapping in real time distributed system. Software mappings often overlooked. This is really surprising because hardware components have much higher reliability than the software that runs over them. Most system designers go to great lengths to limit the impact of a hardware failure on system performance. However they pay little attention to the systems behaviour when a software module fails. There are many different techniques for software mapping(e.g. time out, audits, task

rollback, exception handling, and voting). Most Real time systems must function with very high availability even under hardware fault conditions. The most useful hardware mapping techniques are redundancy and load sharing. For tolerating any fault from the system first we require to detect the fault occurred in the system and then isolating it to the appropriate unit as quickly as possible.

2.2 A Detailed Review of Fault-Tolerance Techniques in Distributed System

A distributed system is a collection of independent computers that appears to its users as a single coherent system. Distributed Computing uses multiple geographically distant computers and solves big and complex task very efficiently. In other words, a distributed system is a collection of independent computers that appears to its users as a single coherent system. Computing power of idle hosts is utilized by distributed computing. Distributed systems offer a better price and performance than mainframes. Computing power can be added in small increments in distributed systems. In this way incremental growth can be achieved. Distributed systems allow many users access to a common computing resource thus provide s resource sharing. Thus it allows many users to share expensive peripherals. It makes human-to-human communication easier. As the size of distributed system is increasing day by day chances of faults are increasing. Mean time to failure is decreasing with increase in size and complexity of distributed system. In large and dynamic distributed system millions of computing devices are working altogether and these millions of computing device are prone to failures. Failures of processors, disks, memory, power, and link failure are some examples of failures. Faults are inevitable in larger and dynamic distributed system. Faults may stop or halt execution of distributed system. It disturbs normal execution and may turn system execution in wrong direction. In air traffic control, distributed disaster system, railways reservation system, internet banking a single fault may lead to huge loss of money and even human lives. In such a situation, inclusion of mapping technique is essential. Mapping Techniques enable systems to perform tasks in the presence of faults. There are high chances that more than one fault may occur in distributed system.

2.3 Mappingin distributed system

A system failure occurs when the system behavior is not consistent with its specifications. A system consists of several components, more the number of components; the more are the things that could be faulty. Since failures are caused by faults, a direct approach to improve the reliability of a system is to try to prevent faults from occurring into a system. This approach is called fault prevention. The other approach is fault tolerance. The goal is to provide service despite the presence of faults in the system. The fault prevention methods focus on methodologies for design, testing and validation; whereas fault tolerant methods focus on how to use components in a manner such that failures can be masked. Here onwards, we will be discussing techniques for building fault tolerant distributed systems. Distributed computing is a field of computer science that studies distributed systems. The term distributed system is used to describe a system with following characteristics: it consists of several computers that do not share a memory or clock; the computers communicate with each other by exchanging messages over a communication network; and each computer has its own memory and runs its own operating system. The resources owned and controlled by a computer are said to be local to it, while the resources owned and controlled by other computers and those that can only be accessed through the network are said to be remote or global.

2.4 Mapping in grid computing: state of the art and open issues

Computational grid consists of large sets of diverse, geographically distributed resources that are grouped into virtual computers for executing specific applications. As the number of grid system components increases, the probability of failures in the grid computing environment becomes higher than that in a traditional parallel computing scenario. Compute intensive grid applications often require much longer execution time in order to solve a single problem. Thus, the huge computing potential of grids, usually, remains unexploited due to their susceptibility to failures like, process failures, machine crashes, and network failures etc. This may lead to job failures, violating timing deadlines and service level agreements, denials of service, degraded user expected quality of service. Thus fault management is a very important and challenging for grid application developers. It has been observed that interaction, timing, and omission faults are more prevalent in grid. Mappings the ability of a system to perform its function correctly even in the presence of faults. The mapping makes the system more dependable. A complementary but separate approach to increase dependability is fault prevention. This consists of techniques, such as inspection, whose intent is to eliminate the circumstances by which faults arise. A failure occurs when an actual running system deviates from this specified behaviour. The cause of a failure is called an error. An error represents an invalid system state that does not comply the system specification. The error itself is the result of a defect in the system or fault. In other words, a fault is the root cause of a failure. However, a fault may not necessarily result in an error; nevertheless, the same fault may result in multiple errors. Similarly, a single error may lead to multiple failures.

2.5 System Diagnosis and Mapping for Distributed Computing System: A Review

Hardware, software and networks cannot be totally free from failures. Mappings a non-functional requirement that requires a system to continue to operate, even in the presence of faults. Distributed systems can be more fault tolerant than centralized systems. Agreement in faulty systems and reliable group communication are important problems in distributed systems. Replication of Data is a major mapping method in distributed systems. Recovery is another property to consider in faulty distributed environments. They contain safety problems in DCS are as Integration Problems, Interaction Problems and Problems of trusted relations. They had also proposed a model of safety in DCS, which support the complex protection of various DCS segments and resource forming the virtual computing environment. The model comprises of protection of data transmission protocol, data transmission, safe communication channels, authentication and transfer/display of safety certificates, access rights to resources delimitation, privacy support of the users, trust in DCS, security control of DCS, security tools of DCS and modern standards in the field of DCS safety as components. Moreover they had mentioned that to reduce the vulnerability level in DCs and to avoid the losses of critical information, additional security mechanisms like security risk analysis in DCS should be taken.

3. Existing System

Existing IT platforms and solutions for big data analytics are designed to operate on large clusters of processing nodes, located in the same data centre (DC). Additionally, these platforms assume the availability of virtually unlimited resources, such as compute power and network bandwidth. When executing big data analytics in telecommunication clouds, however, these assumptions cannot be taken for grante anymore. First, telecommunication clouds tend to be highly distributed in nature, being built up as a constellation of micro DCs in the edge and/or

access network. These micro DCs have the unique benefit to be located much closer to the end-user, which enables e.g. hosting lower latency services and location aware processes. Second, if the data generation velocity is high and/or the size of the events is large, transporting this data over the network to a central DC may consume a significant portion of the available bandwidth, which overlooks that network bandwidth is a scarce and costly resource making the telecom network valuable to end-users. And presents Continuous Hive (soft mapping), a streaming analytics platform tailored for distributed telecommunication clouds. The fundamental contribution of Soft mapping is that it optimizes query plans to minimize their overall bandwidth consumption when deployed in a distributed telecommunication cloud. But existing system can't consider the mapping in distributed environments.

3.1.1 DISADVANTAGES

- There is no mechanism in fault tolerances.
- May lead to lack of service availability due to multiple system failures on multiple failure points.
- As the size increases there comes the challenge to handle the large Volume in big data.

***4.* PROPOSED SYSTEM**

Distributed big data computing provide many storage to the user. Now a day the user level is highly increased to utilize the services in big data computing. In big data computing the major problem vicinity is fault tolerance. Mappings a major concern to guarantee availability and reliability of critical services as well as data transmission. In order to minimize failure impact on the system, failures should be anticipated and handle. Mapping techniques are used to predict these failures and take an appropriate action before or after failures occur. So we propose a mapping mechanism to detect and then recover from failures. Specifically, instead of simply using a query based configuration, we design a trust based method to detect failures in a fast way. Then, a checkpoint based algorithm is applied to perform data recovery. Our experiments shows that our method exhibits good performance and is proved to be efficient. And evaluate the performance of the system using latency and throughput parameters and visualized in distributed environments. Checkpoint is defined as a designated place in a program at
which normal processing is interrupted specifically to preserve the status information necessary to allow resumption of processing at a later time. Check pointing is the process of saving the status information.

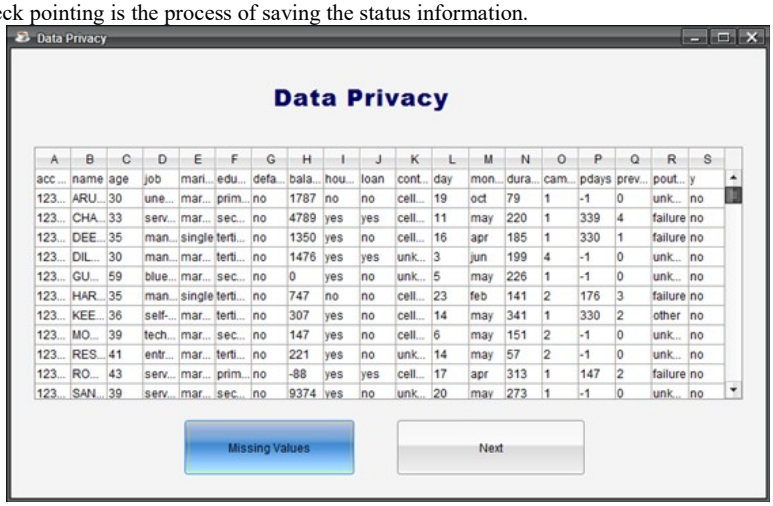


Table 1.1.Data sets

4.1 . Objective of Purposed System

Distributed big data computing provide many storage to the user. Now a day the user level is highly increased to utilize the services in big data computing. In big data computing the major problem vicinity is fault tolerance. Mappings a major concern to guarantee availability and reliability of critical services as well as data transmission. In order to minimize failure impact on the system, failures should be anticipated and handle. Mapping techniques are used to predict these failures and take an appropriate action before or after failures occur. So we propose a mapping mechanism to detect and then recover from failures. Specifically, instead of simply using a query based configuration, we design a trust based method to detect failures in a fast way. Then, a checkpoint based algorithm is applied to perform data recovery. Our experiments shows that our method exhibits good performance and is proved to be efficient. And evaluate the performance of the system using latency and throughput parameters and visualized in distributed environments. Checkpoint is defined as a designated place in a program at
which normal processing is interrupted specifically to preserve the status information necessary to allow resumption of processing at a later time. Check pointing is the process of saving the status information.

4.2 Modeling Map Phase

Cloud computing is sharing of resources on a larger scale which is cost effective and location independent. Resources on the cloud can be deployed by the vendor, and used by the client. It also shares necessary software's and on-demand tools for various IT Industries. In this module, we form the framework with various nodes. The nodes are used to transfer the data from one place to another. Distributed framework is used to communicate the information to various systems.

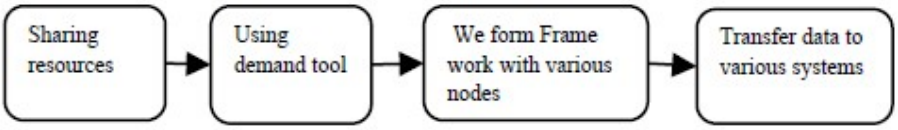
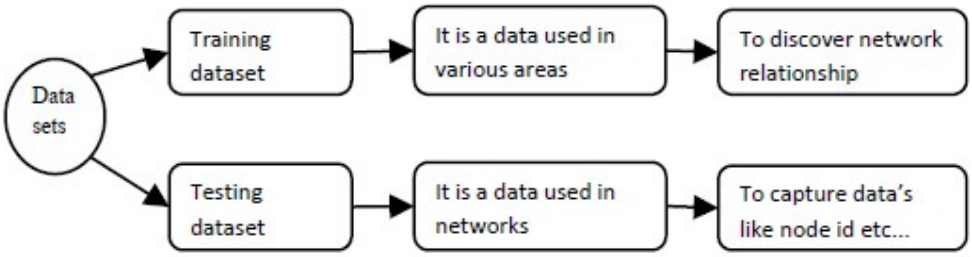


Figure 1.1 .Modeling Mapping

We can create the datasets based data transmission and compared with training datasets. The datasets contains the information such as location details, data details, and attack details and so on. We can create two types of datasets such as training dataset and testing datasets. A training set is a set of data used in various areas of information science to discover network relationship.

Training set has much the same role and is often used in conjunction with a test set. A test set is a set of data used in network to assess the strength and utility of a predictive relationship. Test sets are used in networks to capture the data such as node id, protocol, attacks and so on. The test datasets are classified with training datasets.

Figure:1.2. **Design Rationale**



4.4. Job Execution Estimation

A fault can be categorized on the basis of computing resources and time. A failure occurs during computation on system resources can be classified as: omission failure, timing failure, response failure, and crash failure. In this module predict the faults in distributed systems. Faults may be physical fault or attacker based faults. Fault-tolerance is the important method which is often used to continue reliability in these systems. We can implement the checkpoints are set during the process of job processing, which is overlapped with the failure detection process. That is, while we are monitoring the trust value of nodes, we need to preserve the checkpoint information as well.

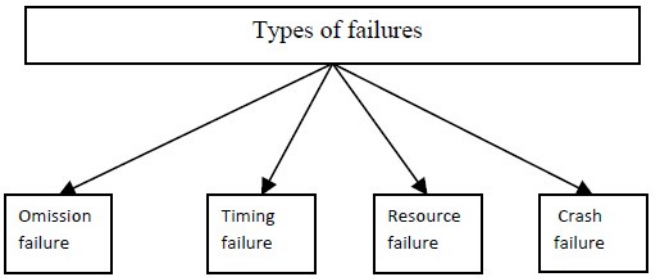


Figure 1.3 Job Execution Estimation

4.5 Resource Provisioning

We can evaluate the performance of the system using latency and throughput measurements. Latency is the delay from input into a system to desired outcome; the term is understood slightly differently in various contexts and latency issues also vary from one system to another. Latency greatly affects how usable and enjoyable electronic and mechanical devices as well as communications are. Latency in communication is demonstrated in live transmissions from various points on the earth as the communication hops between a ground transmitter and a satellite and from a satellite to a receiver each take time. Throughput is a measure of how many units of information a system can process in a given amount of time. It is applied broadly to systems ranging from various aspects of computer and network systems to organizations. Related measures of system productivity

include the speed with which some specific workload can be completed, and response time, the amount of time between a single interactive user request and receipt of the response. Our proposed system provides reduce number of latency and high throughput compared to existing system.

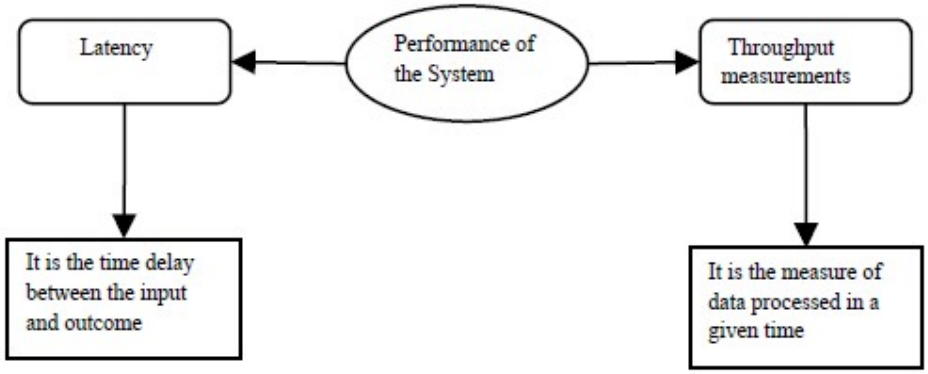


Figure : 1.4 Resource Provisioning

5.1 CONCLUSION

In this paper, we provide a brief introduction of the Map Reduce technique. Most scientific data consists of analyzing huge amount of data collected from different resources. Hence, parallel algorithms and frameworks are the important which can process huge volumes of data and meeting the requirements of performance and scalability entailed in such scientific data analyses. One such popular and efficient framework is Map Reduce which has acquired a lot of attention from the scientific community for its applicability in large parallel data analyses. Even though there are many assessments of the Map Reduce tool employing huge amount of textual data collections, only a few evaluations for scientific data analyses have been made. The explain the Map Reduce programming model as: The calculation takes a group of input key/value pairs, and gives a group of output key/value pairs. The user of the Map Reduce library states the computation as two functions: Map and Reduce. x Map, written by the user, takes an input pair and produces a set of intermediate key/value pairs. The Map Reduce library clusters intermediate values associated with the same intermediate key and passes them to the Reduce function. The Reduce function, also written by the user, accepts an intermediate key and a set of values for that key. It merges together these values to form a possibly smaller set of values. The chapter contains an overview of current big data storage technologies as well as emerging paradigms and future requirements. The overview specifically included technologies and approaches related to privacy and security. Rather than focusing on detailed descriptions of individual technologies a broad overview was provided, and technical aspects that have an impact on creating value from large amounts of data highlighted. The social and economic impact of big data storage technologies was described, and three selected case studies in three different sectors were detailed, which illustrate the need for easy to use scalable technologies. It can be concluded that there is already a huge offering of big data storage technologies. They have reached a maturity level that is high enough that early adopters in various sectors already use or plan to use them. Big data storage often has the advantage of better scalability at a lower price tag and operational complexity. The current state of the art reflects that the efficient management of almost any size of data is not a challenge per se. Thus it has huge potential to transform business and society in many areas. It can also be concluded that there is a strong need to increase the maturity of storage technologies so that they fulfil future requirements and lead to a wider adoption, in particular in non-IT-based companies. The required technical improvements include the scalability of graph databases that will enable better handling of complex relationships, as well as further minimizing query latencies to big datasets, e.g. by using in-memory databases. Another major roadblock is the lack of standardize dint effaces to No SQL database systems. The lack of standardization reduces flexibility and slows down adoption. Finally, considerable improvements for security and privacy are required. Secure storage technologies need to be further developed to protect the privacy of users.

5. FUTURE ENHANCEMENTS

System virtualization is the backbone of Cloud computing, has been liberalizing its services to distributed data-intensive platforms such as Map Reduce and Hadoop. In this paper, we proposed a concurrent VM reconfiguration mechanism for big data tool which is map reduce on virtualized cloud environments Our reconfiguration enhances the input data locality of a virtual Map Reduce cluster. It adds cores to VMs to run local tasks temporarily by scheduling tasks based on data locality, and adjust the computational capability of the virtual nodes to contain the scheduled tasks unlike the traditional schemes which can leads to user-friendly configuration methods for cloud resources. For future work we will consider, create a framework to build relationships between datasets

ACKNOWLEDGMENT

This is to acknowledgement that my paper prepared by myself and future enhancements will be implemented .Those are prove the scientific guidance, and Unpublished results

REFERENCES

[1] Oracle, https://www.oracle.com/database.

[2] P. Larson, C. Clinciu, E. N. Hanson, A. Oks, S. L. Price, S. Rangarajan, A. Surna, and Q. Zhou, "Sql server column store indexes," in SIGMOD, 2011, pp. 1177– 1184.

[3] PostgreSQL, https://www.postgresql.org/.

[4] "Apache hive," https://hive.apache.org/.

[5] Impala, http://impala.io/.

[6] "Vertica," https://www.vertica.com/.

[7] W. A. Giovinazzo, Object-oriented data warehouse design: building a star schema. Prentice Hall PTR, 2000.
[8] Y. Li and J. M. Patel, "Widetable: an accelerator for analytical data processing," PVLDB, vol. 7, no. 10, pp. 907–918, 2014.

[9] D. Serrano and E. Stroulia, "From relations to multidimensional maps: A sql-to-hbase transformation methodology," in AICCSSE, 2016, pp. 156–165.

[10] J.-M. Petit, F. Toumani, J.-F. Boulicaut, and J. Kouloumdjian, "Towards the reverse engineering of denormalized relational databases," in ICDE, 1996, pp. 218–227.

[11] P. Bohannon,W. Fan, F. Geerts, X. Jia, and A.Kementsietsidis, "Conditional functional dependencies for datacleaning," in ICDE, 2007, pp. 746–755.

[12] Z. Khayyat, I. F. Ilyas, A. Jindal, S. Madden, M. Ouzzani,P. Papotti, J.-A. Quian´e-Ruiz, N. Tang, and S. Yin, "Bigdansing: A system for big data cleansing," in SIGMOD, 2015, pp. 1215–1230.

[13] C. R. Carlson, A. K. Arora, and M. M. Carlson, "The application of functional dependency theory to relational databases," The Computer Journal, vol. 25, no. 1, pp. 68– 73,1982.

[14] R. H. Warren and F. W. Tompa, "Multi-column substring matching for database schema translation," in VLDB, 2006, pp. 331–342.

[15] I. F. Ilyas, V. Markl, P. Haas, P. Brown, and A. Aboulnaga, "Cords: Automatic discovery of correlations and soft functional dependencies," in SIGMOD, 2004, pp. 647–658.

[16] P. G. Brown and P. J. Hass, "Bhunt: Automatic discovery of fuzzy algebraic constraints in relational data," in VLDB, 2003, pp. 668–679.

[17] H. Liu, J. Yudian, J. Xiao, H. Tan, Q. Luo, and L. M. Ni, "Ticc: Transparent inter-column compression for columnoriented database systems," in CIKM 2017, to appear.

[18] H. Liu, J. Xiao, X. Guo, H. Tan, Q. Luo, and L. M. Ni,"Cuttle: Enabling cross-column compression in distributed column stores," in APWeb-WAIM 2017, 2017, pp. 219–226.

[19] T. Papenbrock, J. Ehrlich, J. Marten, T. Neubert, J.-P.Rudolph, M. Sch¨onberg, J. Zwiener, and F. Naumann, "Functional dependency discovery: An experimental evaluation of seven algorithms," vol. 8, no. 10, 2015, pp. 1082–1093.

[20] J. Liu, J. Li, C. Liu, and Y. Chen, "Discover dependencies from data—a review," IEEE Trans. on Knowl. And Data Eng., vol. 24, no. 2, pp. 251–264, Feb. 2012.

[21] T. Papenbrock and F. Naumann, "A hybrid approach to functional dependency discovery," in SIGMOD, 2016, pp. 821–833.