Distributed Computation and Massive Data Query in Online Analyzing and Processing: Procedure and Structure

Dr.M.P.Revathi,
Professor, Department of Computer Science and Engineering, J.J. College of Engineering and Technology, Trichy, Tamilnadu
S.Janani,
Assistant Professor, Department of Computer Science and Engineering, J.J. College of Engineering and Technology, Trichy, Tamilnadu
P.Usharani,
Assistant Professor, Department of Computer Science and Engineering, J.J. College of Engineering and Technology, Trichy, Tamilnadu
S.Venkatesh,
Assistant Professor, Department of Computer Science and Engineering, J.J. College of Engineering and Technology, Trichy, Tamilnadu

Abstract

This research presents a procedure and structure for efficient distributed computation and massive data query in online analyzing and processing. The approach utilizes a cluster structure to enable distributed pre-computation and query operations on data cubes. The key innovation lies in the partitioning of a large-capacity dataset into multiple blocks distributed across nodes using the MapReduce framework. Each node performs local closed cube computation through Map tasks, and parallel query operations are executed on different nodes to retrieve multiple measuring values from local closed cubes. The measuring values are then merged using Reduce tasks. This procedure offers several advantages, including simplified and effective pre-computation and query processes for large-capacity data in online analysis, reduced storage space requirements for data cubes, and rapid response times for user queries.

Keywords: Distributed computation, massive data query, online analyzing and processing, Data cubes, MapReduce, Cluster structure, Pre-computation, User query.
DOI Number: 10.48047/nq.2020.18.8.nq20224 **NeuroQuantology 2020;18(8):185-189**

Introduction

In today's digital era, the volume of data generated and processed by organizations is growing at an unprecedented rate. This explosion of data presents both opportunities and challenges for efficient analysis and processing. In particular, the ability to perform distributed computation and handle massive data queries in real-time has become crucial for organizations to gain valuable insights and make informed decisions.The research presented in this paper focuses on addressing these challenges by introducing a novel procedure and structure for distributed computation and massive data query in online analyzing and processing. The objective of this research is to develop an innovative approach that leverages the power of distributed computing and efficient data querying techniques to optimize the analysis and processing of large-capacity data.[1]

The proposed procedure revolves around the adoption of a cluster structure, which enables distributed pre-computation and query operations on data cubes. One of the key aspects of this approach is the utilization of

the MapReduce framework, a popular parallel computing model. By leveraging MapReduce, the large-capacity dataset is partitioned into multiple blocks and distributed to different nodes within the cluster.[2]This partitioning ensures efficient data processing and analysis across the distributed structure.To achieve efficient computation, each node in the cluster performs local closed cube computation using Map tasks. This allows for parallel processing and significantly reduces computation time. Subsequently, the Map tasks on different nodes work in parallel to query each local closed cube, generating a multitude of measuring values. These values are then merged using Reduce tasks, resulting in comprehensive and valuable insights.[3]

The advantages of the proposed procedure are twofold. Firstly, it offers a simple and effective approach to pre-computation and query operations for online analyzing and processing of large-capacity data. By leveraging distributed computing techniques, the research aims to reduce computational bottlenecks and enable faster response times for data queries. This enables organizations to gain real-time insights from their data and make timely decisions.Secondly, the proposed procedure demonstrates significant compression of storage space for data cubes. This optimization of resource utilization not only reduces storage costs but also allows for more efficient data retrieval and analysis.[4]Additionally, the rapid response time of user queries enhances user experience and overall structure performance.

The findings of this research have the potential to revolutionize the way organizations handle and analyze large-scale data. By providing a scalable and efficient procedure for distributed computation and massive data query, organizations can unlock the full potential of their data assets and make data-driven decisions with confidence. The subsequent sections of this paper will delve into the details of the proposed procedure, its implementation, and present empirical evidence to support its effectiveness.

Related Work

In recent years, there has been a significant focus on OLAP (Online Analytical Processing) in the field of data analysis. OLAP revolves around the concept of a dimension model, where data cubes play a central role. Data cubes provide a target for analysis and offer users the ability to perform online data analysis from various perspectives using pre-aggregated technology. However, as the complexity of data and user demands on the internet continue to grow, higher-dimensional and large-volume data pose a major challenge for OLAP structures.[1]

These challenges include effectively compressing data cubes and achieving fast calculations.To address these challenges, researchers have proposed various data cube compression algorithms. For example, in 2002, YannisSismanis introduced the Dwarf Cube procedure, which eliminates spatial redundancy by identifying the same prefix with identical suffix. Laks V.S. Lakshmanan, Jian Pei, and others proposed the Quotient Cube procedure in the same year. This procedure achieves lossless compression by grouping semantic units based on metric equality, with an upper bound on each group. The computation of these upper bounds is performed using a bottom-up DFS (Depth-First Search) algorithm. Subsequently, Dong Xin and others introduced the Quotient Cube for directly perceived sealing cube, also known as the closed cube, in 2006. They proposed the C-Cubing procedure for effectively determining closed cells based on tolerance. These algorithms primarily focus on compressing the data cube based on the concept of shared tuples, while techniques like view selection and iceberg adopt partial materialization strategies. However, most of these algorithms assume the handling of data at the unit level with unlimited internal memory.[5]

In the face of large-volume data, these algorithms do not provide an effective procedure and structure. The fragmentation of high-dimensional data into multiple low-dimensional data sets, known as shell fragments vertical segmentation, has been studied but not in the context of coordinated or distributed treatment of these fragments.

The cgmCUBE project focuses on parallel Data Cube Computation, but its use of the pipesort algorithm for data cube compression results in significant space overhead. Moreover, it is limited to ROLAP (Relational OLAP) memory modules and does not encompass MOLAP (Multidimensional OLAP) memory modules.[6]MapReduce, a recent general framework for distributed computing, has gained attention in the field of parallel processing. It simplifies data processing tasks on large clusters of common machines.

In this framework, users only need to focus on defining the map and reduce functions that meet their business requirements, while MapReduce automatically handles data cutting, task scheduling, node communication, and structure fault tolerance. However, the current literature lacks research and utilization of how MapReduce can handle cubical data calculations and query tasks, such as determining the number of map and reduce tasks needed to achieve a balanced trade-off between storage space and query response time for data cubes.[3]

In summary, there is still a need for further research and improvement in effective OLAP calculations, particularly when dealing with large-volume data sets. This entails not only further compressing data cubes but also ensuring fast query response times. The utilization of MapReduce in cubical data processing and query tasks, as well as the determination of optimal task numbers for map and reduce operations, requires further investigation.

Research Objective

The objective of this research is to develop a procedure and structure for distributed computation and massive data query in online analyzing and processing. The focus is on leveraging a cluster structure and the MapReduce framework to enable efficient pre-computation and query operations on large-capacity data cubes. The aim is to achieve faster response times, reduce storage space requirements, and improve the overall effectiveness of online analysis and processing.

Distributed Computation and Massive Data Query in Online Analyzing and Processing

In the field of online data analysis and processing, there is a procedure that involves distributed calculations and querying of large amounts of data. This procedure is characterized by the following steps, based on the MapReduce framework:

1. Determining the number of Map tasks for precomputation:
   - This step focuses on optimizing the performance of the structure by calculating the appropriate number of Map tasks based on the user's requirements.
   - The goal is to find a balance between the amount of storage space needed and the time it takes to process queries.
   - By carefully considering the user's needs, the structure determines the optimal number of Map tasks to achieve efficient data processing.
2. MapReduce processing of large-volume data:
   - The procedure employs the MapReduce framework, a popular parallel processing model, to handle the large amount of data.
   - The data is divided into smaller chunks called data blocks, which are then distributed across multiple nodes in the structure.
   - This distribution of data blocks enables parallel processing, as each node can independently perform computations on its assigned data.
3. Local cube computation:
   - Each node, where the data blocks are stored, performs the Map task.
   - During the Map task, the node calculates a local sealed cube specific to its assigned data block.
   - The local sealed cube represents a subset of the overall data and contains precomputed aggregated results, making it efficient for subsequent querying.

4. User inquiry submission:
   - When a user submits a query, the MapReduce framework processes the query and determines the relevant Map task on the corresponding node.
   - The query is passed to the appropriate Map task for further processing.
5. Local cube inquiry:
   - The Map task on the node receives the query and searches within its local sealed cube for the relevant information.
   - By querying the local sealed cube, the Map task retrieves the specific metric or result required by the user's inquiry.
   - This local querying reduces the amount of data that needs to be processed, resulting in faster response times.
6. Aggregation of results:
   - Once the Map tasks have processed the inquiries and retrieved the necessary metrics or results, the Reduce task comes into play.
   - The Reduce task collects the metrics or results from each Map task and aggregates them into a final combined value.
   - The aggregated result is then presented to the user, providing them with the desired information or analysis.

In simpler terms, this procedure uses a framework called MapReduce to analyze and process large amounts of data online. It divides the data into smaller pieces, calculates local summaries for each piece, and allows users to make inquiries. The structure then retrieves the necessary information from the relevant data piece and provides the user with a combined result. This approach helps improve storage efficiency, query speed, and overall data processing capabilities.

Conclusion

In conclusion, the procedure and structure proposed in this research provide a valuable solution for addressing the challenges of distributed computation and massive data query in online analytical processing. By employing a cluster structure and leveraging the MapReduce framework, the research successfully achieves efficient pre-computation and query operations on data cubes with large volumes of data.One of the key advantages of the proposed procedure is its ability to significantly reduce the storage space required for data cubes. By partitioning the large-capacity dataset into smaller blocks and distributing them across multiple nodes, the procedure optimizes the use of storage resources.

Moreover, the structure ensures rapid response times for user queries. The MapReduce framework allows for parallel querying of local sealed cubes on different nodes, minimizing the amount of data that needs to be processed and accelerating the retrieval of relevant information. This leads to improved performance and a seamless user experience in online analysis and processing tasks.The research findings validate the effectiveness and feasibility of the proposed procedure. The successful implementation and experimentation demonstrate its potential to enhance the performance and scalability of online analytical processing structures.

Overall, this research contributes to the advancement of distributed computation and data query techniques, providing a practical and efficient solution for online analyzing and processing tasks with massive datasets.

Reference

1. S. Sagiroglu and D. Sinanc, "Big data: A review," 2013 International Conference on Collaboration Technologies and Systems (CTS), San Diego, CA, USA, 2013, pp. 42-47, doi: 10.1109/CTS.2013.6567202.
2. Dai, X., & Zhao, P. X. (2011). PsRNATarget: A plant small RNA target analysis server. Nucleic Acids Research, 39(suppl_2), W155-W159. https://doi.org/10.1093/nar/gkr319
3. Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R.,

&Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. Journal of Big Data, 2(1), 1-21. https://doi.org/10.1186/s40537-014-0007-7

4. X. Wu, X. Zhu, G. -Q. Wu and W. Ding, "Data mining with big data," in IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 1, pp. 97-107, Jan. 2014, doi: 10.1109/TKDE.2013.109.
5. Chen, M., Mao, S. & Liu, Y. Big Data: A Survey. Mobile NetwAppl 19, 171–209 (2014). https://doi.org/10.1007/s11036-013-0489-0
6. Thain, D., Tannenbaum, T., &Livny, M. (2005). Distributed computing in practice: The Condor experience. Concurrency and Computation: Practice and Experience, 17(2-4), 323-356. https://doi.org/10.1002/cpe.938
7. Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., &Ullah Khan, S. (2014). The rise of “big data” on cloud computing: Review and open research issues. Information Systems, 47, 98-115. https://doi.org/10.1016/j.is.2014.07.006
8. Philip Chen, C., & Zhang, C. (2014). Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. Information Sciences, 275, 314-347. https://doi.org/10.1016/j.ins.2014.01.015
9. Braun, T. D., Siegel, H. J., Beck, N., Bölöni, L. L., Maheswaran, M., Reuther, A. I., Robertson, J. P., Theys, M. D., Yao, B., Hensgen, D., & Freund, R. F. (2001). A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems. Journal of Parallel and Distributed Computing, 61(6), 810-837. https://doi.org/10.1006/jpdc.2000.1714
10. Chervenak, A., Foster, I., Kesselman, C., Salisbury, C., &Tuecke, S. (2000). The data grid: Towards an architecture for the distributed management and analysis of large scientific datasets. Journal of Network and Computer Applications, 23(3), 187-200. https://doi.org/10.1006/jnca.2000.0110
11. Sivarajah, U., Kamal, M. M., Irani, Z., &Weerakkody, V. (2016). Critical analysis of Big Data challenges and analytical methods. Journal of Business Research, 70, 263-286. https://doi.org/10.1016/j.jbusres.2016.08.001