Loop Query of Big Data with Low Transmission Cost

MA Yan[a], CHEN Yufeng, KONG Gang, CHEN Suhong

State Grid Shandong Electric Power Research Institute, Jinan, China

[a]yanpony@126.com

Keywords: fetch operation, loop query, big data, network transmission

Abstract. At present, big data environment in electric power field develops. Our platform has a poor efficiency of data analysis and the main bottleneck is disk I/O time of fetch operations. In order to address the problem, we use the characteristics that several tasks usually execute on the same data sets simultaneously, and propose a Loop Query method based on Fetch operations sharing (LQF). It lets all the operations on the data set to process each record circularly, which largely improves the efficiency of data analysis. We also present a Data Placement algorithm with low Transmission cost (DPT). It extends the queries to the cloud environment and reduces the result transmission cost of fetch operations. Last extensive experiments show that the proposed LQF algorithm performs 75.4% better than the benchmark solution when the number of tasks is large, and the DPT algorithm has on average 47.6% less execution time than random placement method.

Introduction

The rapid development of power grid and the explosion of transmission and transformation equipment lead to a rapid increase of amount of data. The data are separately collected into different business information systems during grid operation and equipment monitoring. These systems include production management system (PMS), energy management system (EMS), condition monitoring system of transmission and transformation equipment, geographic information system (GIS) and meteorlogical information system, etc. The environment of electric power big data forms. Big data platform of equipment condition assessment is used to research dynamic evaluation of load capacity, fault diagnosis, condition evaluation and risk assessment [1].

The core value of big data is to store and analyze the massive data. In big data environment, the data engine needs to process huge requests of data analysis from users. Each user has its own timeliness requirements. In the case, the data engine demands the parallel processing mechanism to do the numerous tasks of data analyses, which causes a heavy load stress. Thus, an optimization of parallel processing for big data is necessary to improve the efficiency of data analysis.

In a task of data analysis, there are many fetch operations. The fetch operations of big data do the data I/O from disks, which cost a long time. Each fetch operation occupies many system resources and causes the performance bottleneck. Among the massive tasks of data analysis, there appear several tasks that simultaneously do the fetch operations on one or several data sets, which provide the conditions for sharing the fetch operations. However, sharing the fetch operations has several problems such as read/write conflicts and results returning chaos.

In the paper, we design a Loop Query based on Fetch operations sharing (LQF). It recurrently selects the records in data set and makes multiple fetch operations be shared in order. We also give a Data Placement algorithm with low Transmission cost (DPT) in the cloud. It puts the data sets usually used in comprehensive analysis on one or several neighboring nodes, which reduces the transmission cost of the results of fetch operations and improves the efficiency of data analysis.

Related Work

At present, the tasks of data analysis are processed parallelly in big data environment. About its performance optimization, it is the common method for multiple threads to share a data processing resource. Cuzzocrea et al. [2, 3] propose a framework to support effective and efficient OLAP in big data environment. It combines data partitioning strategy with cloud computing technique to improve query processing. Li et al. [4] present a scalable distributed system to support real-time query by extending the MapReduce framework. It periodically materializes the real-time data into a data cube and compacts the historical versions into one version, which guarantees the response time and throughput. Arres et al. [5] propose a data warehouse placement policy to improve query gain performances on multi nodes clusters. It adopts the existing colocation mechanisms to improve query performances on a Hadoop cluster. Our work in the paper focuses on sharing the fetch operations, which is a higher layer for sharing.

Loop Query Based on Fetch Operations Sharing

In the section, we introduce the application scenario of sharing the fetch operations, and give the implementation method of loop query based on fetch operations sharing.

In big data environment, each user issues several tasks of data analysis to the data engine and the data engine assigns a data processing unit for each task. Let u_i denote a processing unit. There are fetch operations on several data sets in each u_i. Assming that R_j denotes a data set, the fetch operation on R_j of u_i is denoted as $o_{i,j}$. When there are multiple fetch operations on R_j, the sharing of fetch operation can be implemented. As shown in Fig. 1, the steps of the sharing on a data set R_j are as follows.

1) When a processing unit u_i issues a fetch operation $o_{i,j}$, the $o_{i,j}$ operation is added into the operation set on R_j, denoted as O_{R_j}.

2) If O_{R_j} is not null, each record r in R_j is selected in turn.

3) The record r is respectively processed by the query predicate of each operation in O_{R_j}, and the results are then returned to the correponding processing units (denoted as u_i) apart for the further analyses.

4) For any $o_{i,j} \in O_R$, if its once loop query is finished, delete $o_{i,j}$ from O_{R_j}.

When a record is selected, LQF sends it to all the $o_{i,j}$ in O_{R_j}. The approach largely reduces disk I/O operations and the occupation of system resources. The efficiency of data analysis is improved.

Data Placement with Low Transmission Cost in the Cloud

In the cloud computing environment with share-nothing structure, data transmission is one of the most time-consuming steps. A processing unit needs to fetch many data sets. The data sets that always are fetched by the same processing unit are called high-correlation data sets. In the section we introduce a data placement method with low transmission cost in the cloud (DPT). It puts the high-correlation data sets on one or several neighboring nodes to reduce the transmission cost of the results of fetch operations.

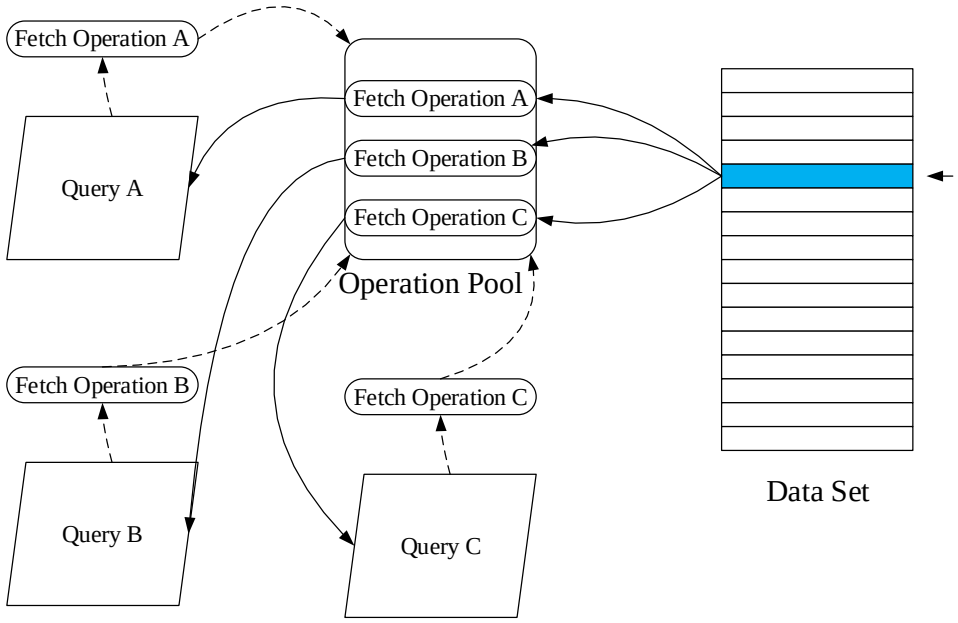

Fig. 1. Query processing process based on polling the data sets

The steps of DPT are as follows.

1) The number of the cloud nodes is denoted as n. The collection of data sets is denoted as $\mathbb{R}$. The number of data sets placed on each node is denoted as m.

2) In a period, the degrees of correlation between all the data sets are performed statistics according to the log of tasks arrival. The concrete method is as follows. Each data set is assumed as a point. If two data sets are simultaneously accessed by a data analysis , an edge weighted by 1 is added between them. The weights of the edge are cumulative, i.e., the weight is added by 1 directly if the edge already exists. All the data sets form a connected graph.

3) An improved clustering algorithm based on K-Means [6] is used to divide $\mathbb{R}$ into n clusters.

4) The number of data sets in each cluster, denoted as d_C, is various. The n clusters are sorted in ascending order of d_C.

5) Each cluster is in turn placed on the nodes in the cloud. If $d_C < m$ holds, multiple clusters are put on a node. The capacity of each node does not exceed m. If $d_C > m$ stands, the cluster is put into the collection of data sets $\mathbb{R}'$ and is split into several parts with each size m.

6) If $\mathbb{R}' \neq \varnothing$ stands, do $\mathbb{R} \leftarrow \mathbb{R}'$ and goto step 3).

Experimental Results

In the section we use the task set of data analysis to validate the performances of both LQF and DPT. A platform with 24 nodes based on Openstack is built as our experimental environment. Each node has 4G memory.

The data processing time on task set of data analysis based on LQF in single-node environment is first validated. The number of tasks in the set ranges from 4 to 64. Assume that the tasks in the set are issued simultaneously and each task does the fetch operation on the same data set. The comparison object of LQF is the case when normally executing the task set in parallel, called as parallel query. As shown in Fig. 2, LQF has an obvious advantage when the number of tasks is large. This is because LQF just scans once data sets and parallel query has many I/Os. When the number of tasks is small, the time of scanning once data sets is more than the I/O time of just executing a small number of tasks. That's why LQF takes more time at the beginning. Moreover, the execution time of LQF does not increase largely with the ascending number of tasks, since the I/O step is the main time-consuming part in data analysis and its time depends on the time of scanning once data sets.

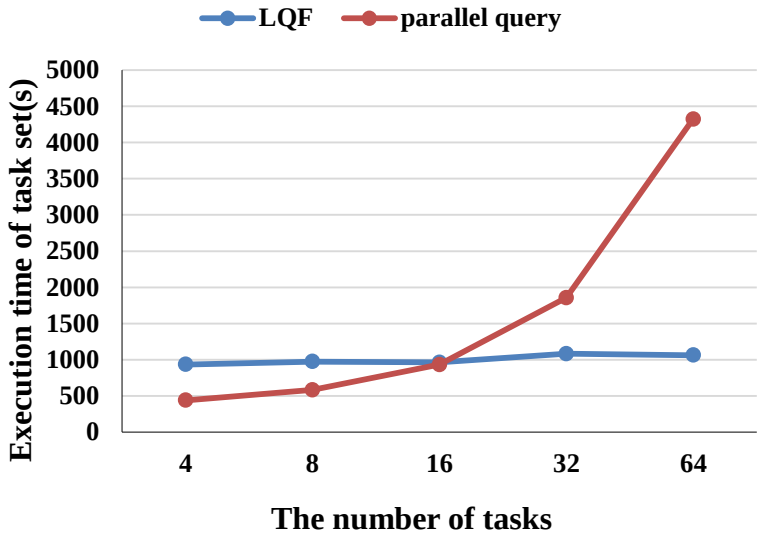

Fig. 2. Data processing time on data sets with different number of tasks

We next demonstrate the execution time of task set based on DPT in the cloud. Assume that the number of tasks is 16 and each task does data analyses on 3 to 4 data sets. A random placement method is compared with DPT. Random placement method is to put the data sets randomly on the nodes. As seen in Fig. 3, DPT takes shorter time especially when the number of nodes is large, while random placement takes more time due to high transmission cost.

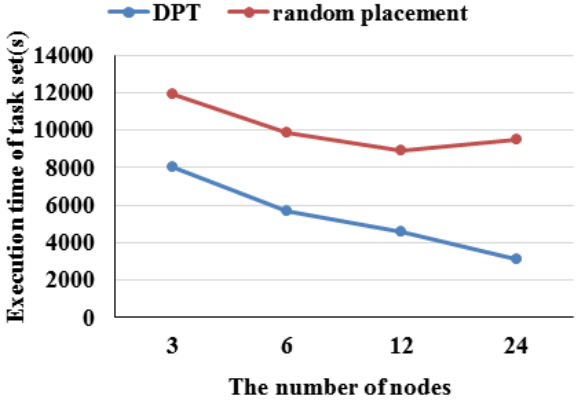

Fig. 3. The execution time of task sets with the number of nodes

Summary

In the paper, we propose a loop query method through sharing the fetch operations called LQF. It optimizes the parallel processing of big data and solves the existent problem of fetch operations sharing. A data placement method called DPT is designed as well to further improve the efficiency of data analysis. It put the data sets with the high degree of correlation on one or several neighboring nodes in the cloud to reduce the transmission cost. Experiments demonstrate that LQF has better performance than parallel query when the number of tasks is large and DPT performs 47.6% better than random placement.

Acknowledgement

This work was supported by the National High Technology Research and Development Program of China (863 Program) under grant no. 2015AA050204.

References

[1]. X.S. Peng, D.Y Deng, S.J Cheng, J.Y. Wen, Z.H. Li and L. Niu： Key Technologies of Electric Power Big Data and Its Application Prospects in Smart Grid, Proceedings of the CSEE, Vol.35(1), pp. 503-511 (2015).
[2]. Alfredo Cuzzocrea and Rim Moussa: A Cloud-based Framework for Supporting Effective and Efficient OLAP in Big Data Environments, the 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, pp. 680-684 (2014).
[3]. Alfredo Cuzzocrea: Analytics over Big Data: Exploring the Convergence of DataWarehousing, OLAP and Data-Intensive Cloud Infrastructures, the 37th Annual Computer Software and Applications Conference (COMPSAC), pp. 481-483 (2013).
[4]. Feng Li, M. Tamer O ¨zsu, Gang Chen and Beng Chin Ooi: R-Store: A Scalable Distributed System for Supporting Real-time Analytics, ICDE Conference, pp. 40-51 (2014).
[5]. Billel Arres, Nadia Kabachi and Omar Boussaid: Optimizing OLAP cubes construction by improving data placement on multi-nodes clusters, the 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, pp. 520-524 (2015).
[6]. Ng R.T. and Han J: Efficient and effective clustering methods for spatial data mining, Proc. 20th Int. Conf. on Very Large Data Bases, pp. 144–155(1994).