ICCSNT 2024

Accelerating Power Big Data Queries by Binary Partitioning and Workload-Driven Pre-joins

Ruoyu Zhao, Tingdong Wang, Yinglong Ma
School of Control and Computer Engineering
North China Electric Power University
Beijing 102206, China

Summary

In this paper, we present a binary partitioning and workload-driven pre-joins approach (BP-WPJ) to enhance the efficiency of cross-departmental big data queries within the power industry. A binary partitioning is utilized to transform the original power property table data into binary table data for effectively filtering out null data and reducing the overall storage requirements. We propose a workload-driven pre-joins processing for reducing the number of repeated data scans and joins, thus improving the efficiency of cross-departmental queries. A query optimization module is used to improve the response speed of queries containing multiple unbound predicates. A large number of experimental results on three datasets show that compared with the established methods such as S2RDF, S3QLRDF, and WORQ, the proposed method in this paper can greatly reduce the execution time of the query statement while reducing the storage space overhead. Evaluation experiments with WORQ on the power dataset show that our method can effectively improve the query response speed for unbound predicate query scenarios. The findings from these experiments validate the effectiveness of BP-WPJ.

Framework

The overall framework of our BP-WPJ approach is shown in Fig 1. The whole framework consists of two main parts: offline data preprocessing and online querying.

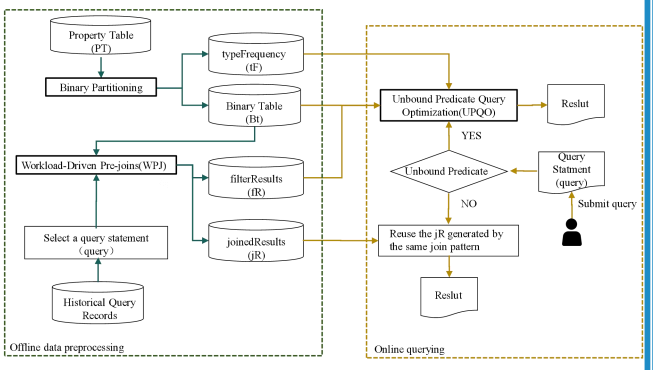

Fig. 1. Average query execution time

Experiments

In order to evaluate the effectiveness of the data optimization module in this paper's approach to alleviate the memory storage space waste problem, we compare the memory storage space consumption of BP-WPJ with other different approaches. The experimental results are shown in Fig. 2.

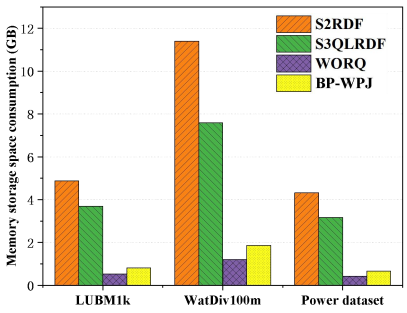

Fig. 2. Memory storage space consumption

We run twenty different query statements on different datasets and calculate their average running times to evaluate the advantages of this paper's approach in data query acceleration. The experimental results are shown in Fig. 3.

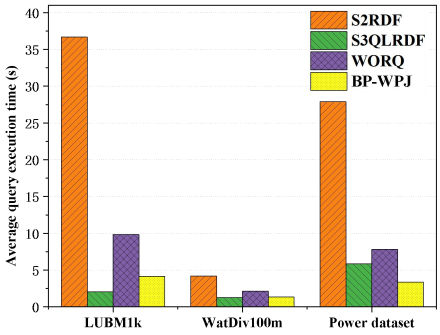

Fig. 3. Average query execution time

We make the experimental methods execute queries containing multiple unbound predicates and compute the average execution time.The results are shown in Fig. 4.

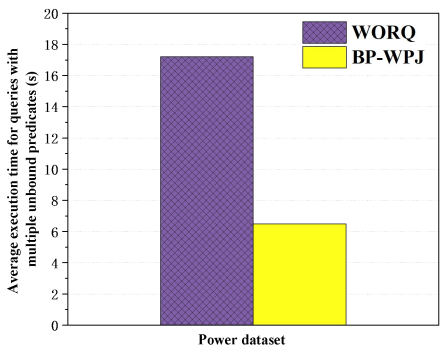

Fig. 4. Average execution time of queries with multiple unbound predicates