Evaluation of Query Modifications in Progressive Data Processing

Nantia Makrynioti - CWI

Abstract

In recent years we have seen an increasing interest in interactive data exploration and visualization applications due to the large volume and heterogeneity of data. Users typically interact with these applications through interface components following a trial-and-error approach, where they submit different queries and based on the results, they refine these queries further. As a consequence, such applications call for a progressive processing paradigm, where periodical feedback is returned to the user, instead of the traditional execution model, where a complete answer needs to be computed.

Data from a quite recent user study [1] on interactive visualization applications show that although real-time interactions pose performance challenges to underlying database systems, the queries generated by many of these interactions present significant similarities. The reason for this is that these queries are usually the result of a steering action on a running query through mechanisms like sliders, panning, adding attributes etc, creating in this way a modified version of the running query. Typical RDBMSes treat queries independently. Recycling of intermediate results, as proposed in previous work [2], mainly targets traditional query processing and assumes reusing of well-defined state or complete results. We examine the sharing of incomplete state/results between relevant queries in the context of the progressive processing paradigm, in order to evaluate such queries more efficiently avoiding redundant work. Two questions that naturally arise are the handling of incomplete state/results when an interruption happens during query execution, as well as how to communicate query modifications to the system. For the former we propose evaluation strategies that map each query modification to a reusable state between the initial and the altered version of a query, and change the query plan accordingly to take the reusable state into account. For the latter we elaborate on the idea of the ALTER QUERY command discussed in [3] and design a SQL extension as a means to communicate modifications on specific parts of a query to the RDBMS.

Preliminary experimental results on specific query modifications and evaluation strategies demonstrate considerable performance gains.

References

[1] Database Benchmarking for Supporting Real-Time Interactive Querying of Large Multi-Dimensional Data. L. Battle, P. Eichmann, M. Angelini, T. Catarci, G. Santucci, Y. Zheng, C. Binnig, J.D. Fekete, D. Moritz. SIGMOD 2020.

[2] An Architecture for Recycling Intermediates in a Column-store. M. G. Ivanova, M. L. Kersten, N. J. Nes, R. A.P. Gonçalves. SIGMOD 2009.

[3] Progressive Data Analysis and Visualization (Dagstuhl Seminar 18411). J. D. Fekete, D. Fisher, A. Nandi, M. Sedlmair. Dagstuhl Reports. 2019.