Adapting to Changing Resource Performance in Grid Query Processing

Anastasios Gounaris[1], Jim Smith[2], Norman W. Paton[1], Rizos Sakellariou[1], Alvaro A.A. Fernandes[1], and Paul Watson[2]

[1] University of Manchester
{gounaris, norm, rizos, alvaro}@cs.man.ac.uk
[2] University of Newcastle upon Tyne
{jim.smith, paul.watson}@ncl.ac.uk

Abstract. The Grid provides facilities that support the coordinated use of diverse resources, and consequently, provides new opportunities for wide-area query processing. However, Grid resources, as well as being heterogeneous, may also exhibit unpredictable, volatile behaviour. Thus, query processing on the Grid needs to be adaptive, in order to cope with evolving resource characteristics, such as machine load. To address this challenge, an architecture is proposed that has been empirically evaluated over a prototype Grid-enabled adaptive query processor instantiating it.

1 Introduction

Grid query processing is particularly relevant where there is a need to integrate information and analysis from different sources for specific periods of time, and to *e-Science* applications, the owners of which, contrary to the typical *e-business* scenario, lack the computational capacity to run some of their tasks and conduct *in-silico* experiments. Especially for the latter case, Grid query processing, like many Grid computations, is likely to place a significant emphasis on high-performance and scalability. Traditionally, query processors often attain scalability and improved performance by relying on the benefits of parallelism. *Pipelined* parallelism has been examined and adopted to different extents in wide-area query processing [15]. Complementarily, query processing can benefit significantly from partitioning the operators within a query execution plan across multiple nodes, a form of parallelism commonly referred to as *intra-operator* or *partitioned* [13], in which all the clones of an operator evaluate a different portion of the same dataset in parallel. GridDB [16] and OGSA-DQP [1] are examples of Grid-enabled database systems that support access to Grid computations and databases, and exploit parallel heterogeneous infrastructures to meet demanding application requirements.

A basic difficulty in efficiently executing a query on the Grid is that the unavailability of accurate statistics at compile time and evolving runtime conditions (such as CPU loads and network bandwidth) may cause load imbalance that detrimentally affects the performance of static techniques for partitioned

J.-M. Pierson (Ed.): VLDB DMG 2005, LNCS 3836, pp. 30–44, 2005.

parallelism. Hence, a challenge for the query processor is to define intra-operator data-partitioning that takes into account these changes. Failing to do so in an efficient way may annul the benefits of parallelism. Just as in homogeneous, controlled environments (e.g., clusters of similar nodes), a slowdown in even a single machine that is not followed by the correct rebalancing, causes the whole system to underperform at the level of the slow machine [2]. To tackle this, the system needs not only to be able to capture these changes as they occur in a wide-area environment, but also to respond to them in a comprehensive, timely and inexpensive manner by devising and deploying appropriate repartitioning policies.

Adaptive load balancing becomes more complicated if the parallelised operations store intermediate state or have incoming queues, like the *hash join* and *exchange* query operators (we call such operators stateful). Assume, for example, that a query optimizer constructs a plan in which there is a hash join parallelised across multiple sites. A hash function applied to the join attribute defines the site for each tuple. In this case, any data repartitioning of unprocessed tuples needs to be accompanied by repartitioning of the hash tables that had already been created within the hash joins.

This paper presents a comprehensive, effective and efficient solution to the problem above. It dynamically rebalances intra-operator parallelism across Grid nodes for both stateful and stateless operations and, in particular, it makes the following contributions:

- It proposes an architecture for *adaptive query processing* (AQP) that is characterised by the following features: it is non-centralised, it is service-oriented, and its components communicate with each other asynchronously according to the publish/subscribe model. Thus it can be applied to loosely-coupled, autonomous environments such as the Grid.
- It presents an implementation of the architecture through extensions to the OGSA-DQP[1] distributed query processor for the Grid [1], demonstrating the practicality of the approach. The resulting prototype has been empirically evaluated and the results show that it can yield significant performance improvements, in some cases by an order of magnitude, in representative examples. In addition, the overhead remains reasonably low, which is important when adaptivity is not required.

The remainder of the paper is structured as follows. The extensions to the static OGSA-DQP system in order to transform it into an adaptive one are presented in Section 2. Section 3 demonstrates adaptations to workload imbalance. Related work is in Section 4, and Section 5 concludes the paper.

2 Grid Services for Adaptive Query Processing

OGSA-DQP has been implemented over the Globus Toolkit 3 Grid middleware. It provides two types of Grid Services to perform static query process-

[1] OGSA-DQP is publicly available in open-source form from www.ogsadai.org.uk/dqp.