Chapter 6.13
Optimal Service Ordering in Decentralized Queries Over Web Services

Efthymia Tsamoura
Aristotle University of Thessaloniki, Greece

Anastasios Gounaris
Aristotle University of Thessaloniki, Greece

Yannis Manolopoulos
Aristotle University of Thessaloniki, Greece

ABSTRACT

The problem of ordering expensive predicates (or filter ordering) has recently received renewed attention due to emerging computing paradigms such as processing engines for queries over remote Web Services, and cloud and grid computing. The optimization of pipelined plans over services differs from traditional optimization significantly, since execution takes place in parallel and thus the query response time is determined by the slowest node in the plan, which is called the bottleneck node. Although polynomial algorithms have been proposed for several variants of optimization problems in this setting, the fact that communication links are typically heterogeneous in wide-area environments has been largely overlooked. The authors propose an attempt to optimize linear orderings of services when the services communicate directly with each other and the communication links are heterogeneous. The authors propose a novel optimal algorithm to solve this problem efficiently. The evaluation of the proposal shows that it can result in significant reductions of the response time.

INTRODUCTION

Technologies such as grid and cloud computing infrastructures and service-oriented architectures have become adequately mature and have been adopted by a large number of enterprises and organizations. This trend has altered, to an extent, the way complex computational tasks are formulated, giving rise to approaches that rely on composition of services to be executed in a parallel and distributed manner (Alpdemir et al.,

DOI: 10.4018/978-1-4666-0879-5.ch6.13

2004; Ng, Ooi, Tan, & Zhou, 2003; Malik, Szalay, Budavari, & Thakar, 2003). As a consequence, there is a growing interest in systems that are capable of processing complex tasks formulated as Web Service (WS) workflows utilizing remote computational resources. These tasks may involve the complete management of online sales, the cleaning of large volumes of accumulated data from mistypes, incorrect entries, etc., and the loosely-coupled integration of local applications with tools made available on the Web.

In Srivastava, Munagala, Widom, and Motwani (2006), the notion of Web Service Management System (WSMS) is introduced as a general purpose system, which possesses such advanced processing capabilities. In a WSMS, processing of data takes place through (remote) calls to WSs. The latter provide an interface of the form *WS*: $X \rightarrow Y$, where X and Y are sets of attributes, i.e., given values for attributes in X, *WS* returns values for the attributes in Y, as shown in the following example adapted from Srivastava et al. (2006). In the generic case, the input data items (or tuples) may have more attributes than X, while attributes in Y are appended to the existing ones. Note that in the rest of this article, we will use the terms tuple and data item interchangeably.

Example 1. Suppose that a company wants to obtain a list of email addresses of potential customers selecting only those who have a good payment history for at least one card and a credit rating above some threshold. The company has the right to use the WSs listed below (Figure 1) that may belong to third parties. The input data containing customer identifiers is supplied by the user.

There are multiple valid orderings to perform this task, although there is one precedence constraint: WS_2 must precede WS_3. The optimization process aims at deciding on the optimal (or near optimal) ordering under given optimization goals. Two possible WS linear orderings that can be formed using the above services are $C_1 = WS_2\ WS_3\ WS_1\ WS_4$ and $C_2 = WS_1\ WS_2\ WS_3\ WS_4$. In the first ordering, first, the customers having a good payment history are initially selected (WS_2, WS_3), and then, the remaining customers whose credit history is below some threshold are filtered out (through WS_1). The C_2 linear plan performs the same tasks in a reverse order. The above linear orderings have different response time. In a subsequent section it will be shown that C_2 is the optimal one.

Optimizing the order of WS calls in a workflow is an important problem that arises in many business and e-science problems. Another example taken from bioinformatics is presented in Craddock, Lord, Harwood, and Wipat (2006), where, given a set of proteins taken from twelve Bacillus bacterium species, the goal is firstly to classify the secreted proteins and then to analyze them through clustering. The analysis of Bacillus bacteria is crucial not only due to their industrial usages in the production of enzymes and pharmaceuticals, but also because of the diversity of their exposed characteristics; the Bacillus genus includes species that are capable of promoting plant growth and producing antibiotics, as well as harmful bacteria such as the Bacillus anthracis. The goal of analysis is to identify families of Bacillus bacteria with respect to the proteins that they synthesize. In such a workflow, when the order of the performed

Figure 1. Services of example 1

WS_1: SSN id (ssn, threshold) → credit rating (cr)

WS_2: SSN id (ssn) → credit card numbers (ccn)

WS_3: card number (ccn,good) → good history (gph)

WS_4: SSN id (ssn) → email addresses (ea)