Available online at www.sciencedirect.com

ScienceDirect

J. Parallel Distrib. Comput. 68 (2008) 840–854

Journal of Parallel and Distributed Computing

www.elsevier.com/locate/jpdc

Scheduling optimization in coupling independent services as a Grid transaction

Haitao Yang[a,*], Zhenghua Wang[b], Qinghua Deng[a]

[a] Institute of Computing Technology of Chinese Academy of Sciences, Graduate University of Chinese Academy of Sciences, Beijing 100080, China
[b] National Laboratory for Parallel and Distributed Processing, National University of Defense Technology, Changsha 410073, China

Received 13 June 2007; received in revised form 24 January 2008; accepted 28 January 2008
Available online 4 March 2008

Abstract

Due to the dynamic properties of autonomous resource providers, the coupling of independent services as a Grid transaction may abort with inconsistency. In many situations people would resort to compensation actions to regain consistency; consequently there comes the issue of compensation-cost. To handle such an issue, for the first time we set up a costing model for the all-or-nothing transaction of Grid services, and introduce the *ECC* metric to evaluate related service scheduling. The analysis of *ECC* estimation is based on the so-called CC-PreC commit pattern, which is an abstract of a category of common use cases of commit handling. Our analysis theoretically illustrates the high degree of computational complexity of scheduling optimization with respect to the cost labeling, timing and order of requests. Under certain typical conditions we prove that infinite possible schemes of scheduling can be reduced down to a finite set of candidates of scheduling. Especially based on the *ECC* metric, the *caution* scheduling is thoroughly investigated, which as a basic policy could be employed in certain common scenarios, and under which the intuitive *product-first* or *cost-first* schemes are justified in several typical situations.

Keywords: Grid services composite; Scheduling optimization; Expected compensation-cost; All-or-nothing transaction; CC-PreC commit pattern; Costing model; Timeout

1. Introduction

With the evolvement of Grid applications, Grid scheduling problems attract more and more people seeking optimal approaches for fulfilling their work. At the same time, the increasing demands for a transaction guarantee from a variety of Grid computations are raising concerns. However, dynamic and autonomous behaviors of the underlying Grid resource providers might make scheduling problems in transaction processing more subtle or sophisticated. Challenging questions might arise [1] like this one: given that all jobs of the work are supposed to satisfy some constraints such as the overall atomicity, how can we best schedule the work onto different resources,[1] so that we cater to the constraints whilst *trying to minimize expenditure or maximize efficiency*? Furthermore, by what metrics do we measure the expenditure or efficiency? In this paper, we try to explore similar issues with emphasis on compensation-cost analysis. More precisely, the primary constraint herein is to couple necessary independent Grid services on demand as an all-or-nothing Grid transaction, under which this work is devoted to providing a theoretical basis to seek an appropriate scheduling for lower compensation-cost or shorter time-span. Since the issue of optimizing scheduling for lower compensation-cost is still lacking systematic investigation in the service computing fields, we start with basic models and notions, and then try to identify and solve typical problems. The calculability, complexity, and workability of scheduling schemes, and trade-offs are of major interest.

For a better understanding, we outline our contributions first: (a) a formal pattern for time-related composite transactions; (b) a formal model for pricing the compensation-cost of aborted transactions; (c) metrics and a framework of probabilistic analysis for evaluating scheduling.

* Corresponding author.
E-mail addresses: yhtyxc@hotmail.com, yanght@gdcic.net (H. Yang).

[1] Grid resources should include services and should be served through appropriate service interfaces.

0743-7315/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jpdc.2008.01.004

The rest of this paper is organized as follows. First, related researches are reviewed in Section 2. The terminology and conditions are given in Section 3, and subsequently the underlying CC-PreC commit pattern is introduced and explained in Section 4. After basic models for scheduling analysis are prepared in Section 5, two basic schedulings, i.e., the *caution* and the *efficiency-first*, and the computation of their *ECC* are presented in Section 6 while the time-metric analysis of these schedulings is given in Section 7. In-depth discussions of scheduling optimization on *ECC* are developed in Section 8. Illustrative numerical examples are included in Section 9. After the analysis of computational complexity in Section 10, the paper is concluded in Section 11.

2. Related work

On the level of business application of software, the issue of compensation-cost is normally associated with composite transactions. There has been a lot of literature on composite transactions [3,2,7,19] since the 1990s, and early researches on compensatable transactions can be traced back to three decades [6]. Published related work includes articles on the mechanisms to execute or utilize a compensating transaction [14], on the criteria of compensation soundness [7], on adaptive commit protocols for setting local compensatory actions [13], etc. In addition, literature on the timeout-related issues of transaction processing is also available in publications [9,8] though not common outside the field of real-time systems. To the best of our knowledge, however, explicit and quantitative explorations of transaction scheduling on compensation-cost from the point of view of economics are rare. Here, we can only mention several partially-related ones: (1) in 2005, Taylor et al. [15] discussed the cost-quoting facility which is prior to customers' acceptance of services and can be utilized by the value-added service provider to minimize total cost, when they developed charging mechanisms within SOAs that permit dynamic composition of services to achieve customer goals; (2) in 2006, Ghafoor et al. [4] studied the practice of using π-calculus with compensation and exception handling to reduce the scope of services to be compensated, and hence decrease costs; (3) in 2006, Ye et al. [18] presented a model for publishing and discovering services with atomicity sphere for B2B collaboration to avoid compensation, since they considered the cost of compensation expensive. These works tell us that the cost of service combination is of general concern, and the cost of compensation is often of particular interest, which motivate us to carry out a more thorough study on the compensation-cost estimation of transaction process of Grid service combination. Even in the narrow research scope that is overlapped by these related researches and our study, the following aspects still sufficiently distinguish our work from others: firstly, they commonly considered compensation-cost as execution overhead like CPU time, storage usage, etc., that is, their researches are normally developed on the service implementation level [11], while we investigate on the level of pricing economics; secondly, our costing model does not exclude uncompensatable component services, since their direct compensation-costs can be considered equal to their total service price; finally, we focus on the compensation-cost[2] issue since the cost of normal service will be equivalently exchanged with the value of the service required (decided by the market of supply and demand), whereas the compensation-cost is clearly extra and unwanted.

3. Terminology and conditions

In our opinion, Grids can be comprehended on two levels: on the concept level and on the IT practice level. The final goal of introducing Grids is targeting at IT practices. For the first understanding level, although the term Grid has been used in many different ways, basically its references can be classified into two categories: (1) as an abstraction concept it refers to the computing pattern that is featured by that its implementation "coordinates resources that are not subject to centralized control", "uses standard, open, general-purpose protocols and interfaces", and "delivers non-trivial qualities of service", as rendered by [23]; (2) as an indication for a system that can be viewed as an instance of (1). Regarding practical applications, the second usage of the term Grid is more often adopted.

On the IT practice level, we recommend the WS-Resource Frame as the standards or conventions to implement Grids, since it has many advantages over the OGSI line [20,24]. In this paper, we are interested in the Grid that sits on Web services [20] and provides Grid services on demand to carry out *all-or-nothing* transactions for customers who intend to use several Web services together. The Grid service that offers all-or-nothing transactions can be regarded as a WS-Resource, which is the composite of Web services and stateful resources. Stateful resources here refer to the data which records the processing status of Web services and their processed objects (e.g., telling that whether a flight seat is occupied or still vacant). The construction standards and conventions of a Grid service are defined by the specifications within the WS-Resource Frame.

Since the feasibility of Web services depends implicitly on the high quality of networks, the availability of service registries, and the convenient user-authentication mechanism for customers, we employ the following premises here: (a) the network transfer delay is trivial; (b) all necessary Web service description information is published in a public service registry; (c) user-authentication across different autonomous Web services is not our concern; (d) the message transfer is reliable in the Grid.

Further, we classify fundamental technical terms and conditions into the following list:

- *Service* here refers to an execution process of *Grid service*, which is a self-describing, self-contained and modular application accessible over the Grid. It exposes an XML interface, which is registered and can be located through

[2] The compensation-cost of interest in this paper is limited to the direct compensation-cost.