A Scheduling Algorithm for the Distributed Student Registration System in Transaction-Intensive Environment

Wenhao Li, Swinburne University of Technology, Australia

ABSTRACT

Distributed workflow technology has been widely used in modern education and e-business systems. Distributed web applications have shown cross-domain and cooperative characteristics to meet the need of current distributed workflow applications. In this paper, the author proposes a dynamic and adaptive scheduling algorithm PCSA (Pre-Calculated Scheduling Algorithm) for a distributed student registration system in a transaction-intensive environment. In this algorithm, every workflow application is pre-calculated before deployed into the workflow engine to create priority lists for solution generating. The simulations illustrate that the algorithm has a preferable efficiency and fits the needs of the distributed student registration system.

Keywords: 
Distributed Student Registration System, Distributed Workflow, Scheduling Algorithm, Transaction-Intensive, Workflow Technology

INTRODUCTION

Distributed workflow technology is always considered to be a very important part of the workflow research. It decomposes a single workflow into several pieces and executes them on different execution nodes, and transfer parameters between execution nodes to guarantee data integrality and execution orders. At the meanwhile, scheduling is a methodology that organizes all the workflow pieces work together in a certain order, with certain logic. Due to the heterogeneity and some other factors of execution nodes, how to schedule a distributed workflow system in a reasonable way has become a significant field in distributed workflow management system research.

At present, distributed web applications has shown a cross-domain and cooperative character. The implementation of business processes by using workflow and service technology has become a trend. For example, the new generation service-oriented network infrastructure such as Grid and Cloud computing platform, and a variety of applications developed based on them, are all representative of this type (Benkner, Brandic, Engelbrecht, & Schmidt, 2004; Vecchiola, Chu, & Buyya, 2009; Vouk, 2008).

DOI: 10.4018/jdet.2011010105
The growing demand for network services has led to an increasing amount of network accesses. When a big amount of users access to a workflow management system simultaneously, extensive web requests will generate a great number of workflow instances and eventually form a transaction-intensive environment. The distributed student registration system is a typical transaction-intensive workflow system: In a typical university, the student registration system deals with the registration concerns of students, which include registration in courses, change of timetable, withdrawal of courses, etc. The system may experience a heavy load from time to time. Thousands of students, new or current, may lodge their registration requests just before the deadline. Therefore, the performance of the registration processing system clearly is a major concern. The system should be capable of handling a large amount of requests in a short period of time (Yan, Yang, & Raikundalia, 2006).

In this paper, we put forward a dynamic and adaptive scheduling algorithm to meet the need of the distributed student registration system referred above and other distributed systems with similar transaction-intensive characters. The algorithm is named PCSA (Pre-calculated Scheduling Algorithm). In this algorithm every workflow application will be pre-calculated before deployed into the workflow engine to analyze possible instances of the application. At this step each atom activity will generate a priority list for all the available execution nodes. Priorities of the execution nodes represent the “distance” of the node from the optimum one. The activities will always be scheduled on the execution node close to the optimum one, which means that the (activity, node) pair is a part of the solution close to the optimum solution, so that the ultimate solution of each instance is close to the optimum one, too. In this way the algorithm can guarantee that each workflow instance consumes the minimum execution cost and transfer cost, which is either time cost or monetary cost. In this paper the cost only stands for the time the process takes for execution.

We also lay out the simulation tests we did on the distributed student registration simulation system. We compared PCSA with the Random algorithm and the Myopic algorithm (Wieczorek, Prodan, & Fahringer, 2005), the former randomly schedules ready activities on free execution nodes and the latter schedules ready activities on free execution nodes which have the lowest execution cost and transfer cost. In the test we found out that on the aspect of average turnaround time and average execution speed, PCSA have a better performance than Random algorithm and Myopic algorithm.

The rest of this paper is organized as follows: Related work and some background research is discussed. Then we put forward the workflow model. The algorithm PCSA is described in detail and the experiment design and performance of proposed algorithm is examined. At last, we summarize and lay out the future work.

Related Work

Scheduling is explained to be “a process that maps and manages execution of interdependent tasks on distributed resources. It introduces allocating suitable resources to workflow tasks so that the execution can be completed to satisfy objective functions specified by users” (Yu & Buyya, 2007).

Scheduling in distributed systems is NP-complete in general. There are many domain specific algorithms aiming at finding good though not optimal solutions for scheduling (Dong & Akl, 2006). According to the time that scheduling solution is generated, scheduling algorithms can be mainly divided into two groups: static and dynamic. A static algorithm presumes that the whole structure of a workflow application and its related information is aware by the system, and resource mapping is made on DAG level before the execution. Conversely, a dynamic algorithm makes a decision only when an individual job is ready to execute (Yu & Shi, 2008). Both static and dynamic algorithms have their advantages and disadvantages: static
Related Content

Computer-Assisted Instruction: A Case Study of Two Charter Schools
[www.igi-global.com/article/computer-assisted-instruction/76317?camid=4v1a](www.igi-global.com/article/computer-assisted-instruction/76317?camid=4v1a)

The Effects of Web-Enabled Self-Regulated Learning and Problem-Based Learning with Initiation on Students’ Computing Skills
[www.igi-global.com/chapter/effects-web-enabled-self-regulated/50175?camid=4v1a](www.igi-global.com/chapter/effects-web-enabled-self-regulated/50175?camid=4v1a)

Corporate - Higher Education Alliance
[www.igi-global.com/chapter/corporate-higher-education-alliance/11798?camid=4v1a](www.igi-global.com/chapter/corporate-higher-education-alliance/11798?camid=4v1a)
Build an Online Program Consortium
www.igi-global.com/chapter/build-online-program-consortium/11757?camid=4v1a