Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 125 (2018) 935–943

Procedia Computer Science

www.elsevier.com/locate/procedia

6th International Conference on Smart Computing and Communications, ICSCC 2017, 7-8 December 2017, Kurukshetra, India

Time Cheating in Divisible Load Scheduling: Sensitivity Analysis, Results and Open Problems

Shamsollah Ghanbari[a,b,*], Mohamed Othman[a,c,*]

[a]*Institute For Mathematical Research, Universiti Putra Malaysia 43400 UPM Serdang, Selangor D.E., Malaysia*
[b]*Department of Computer Science, Islamic Azad University, Ashtian Branch, Iran.*
[c]*Department of Communication Tech and Network*
Universiti Putra Malaysia 43400 UPM Serdang, Selangor D.E., Malaysia

Abstract

There is broad literature in relation to the Divisible Load Scheduling (DLS). The DLS is based on the fact that the computation and communication can be separated into some arbitrary independent parts, in which each part can be processed by an independent processor. The existing DLS models supposes that the processors report their true communication and computation rates to the root processor. It means the traditional DLS assumes that the processors do not cheat the root processor. In fact, in the real applications, the processors may cheat the algorithm it means the processors might not report their true computation or communication rates. However, the computation rate-cheating problem has been studied in several research. This paper reviews the research in relation to the communication and computation rate-cheating problems. The paper also investigates the sensitivity analysis of the DLS where the processors cheat the root processor for the first time.

Peer-review under responsibility of the scientific committee of the 6th International Conference on Smart Computing and Communications

Keywords: Divisible load scheduling, time cheating, finish time, computation rate-cheating, communication rate-cheating, sensitivity analysis.

1. Introduction

The first published articles about the Divisible Load Theory (DLT) were appeared in 1988, [1, 2]. A class of the scheduling methods which is defined based on the DLT is called the Divisible Load Scheduling (DLS). During the past three decades, the DLS has found a wide range of applications in the area of parallel and distributed computing, such as data grid applications [3, 4], image and vision processing [5, 6] and linear algebra [7]. Moreover, the DLS was performed to diverse topologies, such as single-level tree, multi-level tree, bus and daisy chain [8], three-dimensional

[*] Corresponding authors. Shamsollah Ghanbari & Mohamed Othman Tel:+603-8947 1707
E-mail address: myrshg@gmail.com& mothman@upm.edu.my

1877-0509 © 2018 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the scientific committee of the 6th International Conference on Smart Computing and Communications
10.1016/j.procs.2017.12.119

meshes [9], k-dimensional meshes [10], hypercubes [11] and arbitrary graphs [12]. Moreover, it has been used for scheduling in homogeneous [13] and heterogeneous platforms [14, 15], grid-based environment [16, 17, 18] and cloud-based applications [19].

Recently, some other properties of the DLS, such as multi-layer divisible computations [20], hierarchical memory [21], multi-criteria DLS [22], multi-objective DLS [23] and priority-based DLS [24] have been widely studied. Furthermore, there are some extensive reviews in relation to the DLS concerning the applications, strategies and open issues [25, 26, 27, 28]. The DLS is based on the fact that the load can be partitioned among sections, in a way that each section can be performed by a set of independent processors. The existing DLS supposes that the processors report their true communication and computation rates to the originator, meaning that they do not cheat the algorithm. In fact, in the real applications, the processors may not announce their true computation rates. This issue was studied by Thomas E. Carroll and Daniel Grosu for first time [29, 30, 31, 32]. The results of their investigations demonstrate that the cheating issue degrade the implementation of the DLS model. As a matter of fact, the DLS only achieves its optimal implementation if the processors announce their real computation rates [29, 30]. Few years later, the communication and computation rate-cheating problems have been studied by the other scholars [23, 24]. In [24] a priority-aware DLS model has been developed. That method was able to decrease the impact of communication rate-cheating on the makespan. In [23] an adaptive multi-objective method has been presented. The multi-objective method was able to considerably decrease the impact of computation rate-cheating on the makespan. Although the adaptive DLS method has comprehensive literature, [33, 34, 35, 36, 37] but [23] presented a new approach in the area of the DLS focusing on the misreporting and cheating problem. The present study deeply focuses on the computation rate-cheating problem and analyzes the results of research toward the communication and computation rate-cheating. The main objectives of this paper are:

- to review the research relation to the time cheating in the area of the DLS
- to analyze the sensitivity of the DLS in order to illustrate feasibility of decreasing the effects of the cheating problem in the DLS model

2. Background

2.1. Mathematical Model of the DLS

Basically, the DLS is based on the fact that the computation and communication can be separate into some sections of arbitrary sizes, and each section can be processed independently in parallel. The DLS supposes that, preliminary amount V of load is retained by the root processor denoted by p_0. A common presumption is that, the root processor does not perform any computation. In fact, the root processor only dispatch the load on the worker processors p_1, p_2, ..., p_m. Suppose that the i^{th} processor ($0 \leq i \leq m$) receive α_i fraction of load. According to [38] the condition for getting optimal finish time is that, the processors finish processing simultaneously; under other circumstances, the load can be transmitted from busy to idle processors. In this occasion, the goal is to compute α_0, α_1, ..., α_m in the DLS timing equation.

2.2. Definition of Parameters

In this paper, we assumed that m parallel processors are interconnected to the root processor denoted by p_0. The other processors, denoted by p_1, p_2, ..., p_m are called worker processors. The following parameters have been used throughout this paper.

- w_j: It is the inverse computing speed (computation rate) of the j^{th} processor and can be calculated by $w_j = \frac{\omega_j}{W_j}$, where ω_j and W_j are the time taken by processor p_j and a standard processor respectively to compute a given load.
- z_j: It is the inverse transmission speed (communication rate) of the j^{th} processor and can be calculated by $z_j = \frac{\mu_j}{Z_j}$, where μ_j and Z_j are the time taken by processor p_j and a standard processor respectively to transfer a given load.