The Journal of Systems and Software 86 (2013) 302–314

Contents lists available at SciVerse ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

3E: Energy-efficient elastic scheduling for independent tasks in heterogeneous computing systems

Xiaomin Zhu[a,*], Rong Ge[b], Jinguang Sun[c], Chuan He[a]

[a] Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha 410073, PR China
[b] Department of Mathematics, Statistics, and Computer Science, Marquette University, Milwaukee, WI 53233, USA
[c] School of Electronic and Information Engineering, Liaoning Technical University, Huludao 125105, PR China

ARTICLE INFO

Article history:
Received 7 April 2012
Received in revised form 30 July 2012
Accepted 9 August 2012
Available online 3 September 2012

Keywords:
Heterogeneous computing system
Scheduling
Energy-efficient
Elastic
DVFS

ABSTRACT

Reducing energy consumption is a major design constraint for modern heterogeneous computing systems to minimize electricity cost, improve system reliability and protect environment. Conventional energy-efficient scheduling strategies developed on these systems do not sufficiently exploit the system elasticity and adaptability for maximum energy savings, and do not simultaneously take account of user expected finish time. In this paper, we develop a novel scheduling strategy named energy-efficient elastic (3E) scheduling for aperiodic, independent and non-real-time tasks with user expected finish times on DVFS-enabled heterogeneous computing systems. The 3E strategy adjusts processors' supply voltages and frequencies according to the system workload, and makes trade-offs between energy consumption and user expected finish times. Compared with other energy-efficient strategies, 3E significantly improves the scheduling quality and effectively enhances the system elasticity.

1. Introduction

Heterogeneous computing systems constructed by connecting various machines with different capabilities have been wildly employed for compute-intensive and data-intensive applications in scientific research and commercial industries (Goller and Leberl, 2009; Zheng et al., 2006; Donoho, 2004). The employment is mostly attributed to the high processing capability and low cost of commodity-off-the-shelf hardware components including processors, memory, networks, and storage disks (Xie and Qin, 2008).

Large-scale computing systems consume tremendous amounts of energy, which in turn causes high energy bills in data centers, raises environmental concerns, and increases system failures. The monetary cost for energy is very high. For instance, the total energy cost of a single 200 W server (e.g., IBM 1U*300) is $180/year (Bianchini and Rajamony, 2004). A computing system with thousands of compute nodes or more incurs large energy bills. The high energy consumption also has negative environmental impacts. It is estimated that 1 kWh electricity power requires 0.4 kg coal and 4 L water and produces 0.272 kg solid powder, 0.997 kg CO_2, and 0.03 kg SO_2. Last, high energy consumption results in high temperature that greatly affects the system reliability. According to the Arrhenius equation, the failure rate of an electronic component doubles for every 10 °C increased (Feng, 2003). In order to maintain an appropriate operating temperature in data centers, extra energy will be consumed by the cooling devices and facility.

Dynamic voltage and frequency scaling (DVFS) is an energy saving technology that are enabled on most contemporary processors (http, in pressa; http, in pressb). With DVFS, a processor can operate at multiple voltages where each corresponds to a specific clock frequency and processing speeds. Because the energy consumption of a processor is proportional to voltage squared (Chen et al., 2006), processor's energy consumption can be significantly reduced by lowering CPU voltage and processing speed.

The motivations of this paper derive from the following three considerations:

- *Many applications running on heterogeneous computing systems consist of independent tasks without dependencies.* For example, the tasks submitted to a supercomputer center by different users are independent (Braun et al., 2001); the partitioned data blocks from signal data in a software radio system can be considered as independent tasks without precedence relationship (Zhu and Lu, 2008); a parameter-sweep application consists of a set of independent coarse-grained tasks, and such applications can be seen in diverse areas such as bioinformatics, operations research,

* Corresponding author.
E-mail addresses: xmzhu@nudt.edu.cn (X. Zhu), rong.ge@marquette.edu (R. Ge), sunjinguang@lntu.edu.cn (J. Sun), chuanhe@nudt.edu.cn (C. He).

0164-1212/$ – see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jss.2012.08.017

data mining, business model simulation, massive searches, Monte Carlo simulations, network simulation, electronic CAD, ecological modeling, fractals calculations, and image manipulation (Fujimoto and Hagihara, 2006).

- *There exist many applications for which users do not have strict requirement in finish time.* For these applications, even though the finish time of a task is a bit later or earlier than its user's expectations, the task execution is still useful. Take earth observation satellite as an example. The satellite data are firstly processed at a ground data center in one task, then the generated data are analyzed in the next task for a new electronic reconnaissance. If the first task finishes later than the start time of the second task, the generated data are still useful for another new task (Wang et al., 2010). In the traditional real-time computing systems, a task would be rejected if it is unable to finish within its deadline. For non-real-time applications, only the computing throughput matters while the finish times of single tasks can be ignored. Thereby, in order to provide high-quality services for users, each user's expectation (e.g., expected finish time) should be adequately considered.
- *Emergency tasks and non-emergency tasks show distinct characteristics.* The applications that consist of independent tasks (Xie and Qin, 2008; Doğan and Özgüner, 2006; Mehta et al., 2007; Kang and He, 2009) in data centers can be roughly classified into two groups. One group includes applications for which high performance and short execution time are critical. One example is the processing of image data obtained by earth observation satellites in emergency such as earthquake, tsunami or military operations (Wang et al., 2007). In these scenarios, quick response is paramount while energy savings with possible performance impact are out of the questions. The other group includes applications that are not in emergency in nature. For these applications, we can exploit system elasticity, i.e., flexibility and adaptivity with the variety of system workload, in scheduling algorithms to reduce energy use with no or little impact on meeting user expectations.

Scheduling is an effective approach to achieve high performance and energy efficiency for applications running on heterogeneous computing systems. However, to the best of our knowledge, most existing energy-efficient scheduling algorithms do not address both performance and energy cost for non-real-time applications in heterogeneous computing systems. Motivated by the above arguments, in this work, we attempt to incorporate the system elasticity and user expectations into energy-efficient scheduling strategies, and to design and implement a novel energy-efficient elastic scheduling strategy for non-real-time tasks on DVFS-enabled heterogeneous computing systems. Specifically, our approach firstly strives to satisfy the user expectations by adjusting the execution voltages of queued tasks and new tasks, and then reduces the system energy consumption as much as possible.

Contributions: The main contributions of this paper are:

- We construct an energy consumption model that effectively takes advantage of system elasticity and considers different user expectations in terms of expected finish time.
- On the basis of the novel energy consumption model, we develop an energy-efficient elastic (3E for short) scheduling strategy for independent tasks in heterogeneous computing systems to make trade-offs between energy saving and user expectation according to the system workload.
- We demonstrate that, by considering heterogeneous features of multiprocessor computing systems, we can design an energy-efficient elastic scheduling strategy that significantly improves the scheduling quality of conventional scheduling strategies for heterogeneous computing systems.

The rest of this paper is organized as follows. In Section 2, we discuss the related work in literature. Section 3 presents the system model, task model, energy consumption model. In Section 4, we describe the 3E strategy and its main principles. Section 5 presents the experimental results and performance evaluation. Section 6 concludes the paper with a summary and future directions.

2. Related work

Over the last decades, scheduling in distributed computing context has been intensively investigated. As the optimal scheduling solutions are normally NP-complete (Coffman, 1976), near-optimal solutions using heuristic techniques are adopted as practical alternatives (Xie and Qin, 2008; Braun et al., 2001; Kim et al., 2008; Karatza, 2009).

Scheduling can be generally classified into static scheduling and dynamic scheduling according to its design time. The static scheduling makes scheduling decisions in an off-line planning phase, and is usually used to schedule periodic tasks (e.g., Aydin et al., 2004; Mishra et al., 2003; Yu and Prasanna, 2002; Zhu et al., 2003). The dynamic scheduling is performed in an on-line fashion when tasks arrive at unpredictable intervals, and is usually applied to aperiodic tasks and the system workload that is not known a priori (e.g., Ge et al., 2005; Hu et al., 2008; Zong et al., 2011; Hamano et al., 2009; Zikos and Karatza, 2011; Yan et al., 2005; Zhu et al., 2011; Zhu and Lu, 2009).

Aydin et al. proposed a static solution for periodic tasks to compute the optimal speed at the task level based on the worst-case workload for each arrival (Aydin et al., 2004). Mishra et al. proposed a static power management scheme that used the static slack based on the degree of parallelism in a given static schedule generated from any list scheduling heuristic algorithm (Mishra et al., 2003). Yu et al. investigated an off-line power-aware allocation policy that was firstly formulated as an extended generalized assignment problem, and was solved by an extension of a linearization heuristic for a set of independent tasks in a real-time system consisting of heterogeneous DVS-enabled processing elements (Yu and Prasanna, 2002). Zhu et al. introduced the concept of slack sharing on multiprocessor systems to reduce energy consumption and proposed two novel power-aware scheduling algorithms based on slack sharing for task sets with and without precedence constraints executing on multiprocessor systems (Zhu et al., 2003). Additionally, the study in Zhu et al. (2003) assumed homogeneous processors and frame-based tasks. Tavares et al. proposed a pre-runtime scheduling method that considered the DVS technique to reduce energy consumption and took the inter-task relations and runtime overhead into account. In addition, the time Petri nets was employed as a mathematical basis for precise pre-runtime schedule generation (Tavares et al., 2008). Although these scheduling schemes are capable of achieving high scheduling quality in terms of energy saving, they belong to static scheduling and are unable to deal with dynamic environment where the arrival time of a task is not known.

There is a large body of work in designing dynamic energy-efficient scheduling algorithms for distributed computing systems. Ge *et al.* investigated distributed performance-directed DVS scheduling strategies that could produce significant energy savings without increasing execution time by varying scheduling granularity (Ge et al., 2005). Nélis et al. proposed two power-aware scheduling algorithms, i.e., an off-line algorithm $EDF^{(k)}$ and an on-line algorithm MOTE that both addressed sporadic constrained-deadline real-time systems to reduce energy consumption (Nélis