The Journal of Systems and Software 85 (2012) 906–919

Contents lists available at SciVerse ScienceDirect

The Journal of Systems and Software

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

High performance dynamic voltage/frequency scaling algorithm for real-time dynamic load management

J.O. Coronel*, J.E. Simó**

University Institute of Control Systems and Industrial Computing (AI2), Polytechnic University of Valencia, Camino de Vera s/n, 46022 Valencia, Spain

ARTICLE INFO

Article history:
Received 24 July 2010
Received in revised form 12 October 2011
Accepted 9 November 2011
Available online 18 November 2011

Keywords:
Dynamic voltage scaling
Task migration
Real-time scheduling
Power consumption
Feasibility analysis

ABSTRACT

Modern cyber-physical systems assume a complex and dynamic interaction between the real world and the computing system in real-time. In this context, changes in the physical environment trigger changes in the computational load to execute. On the other hand, task migration services offered by networked control systems require also management of dynamic real-time computing load in nodes. In such systems it would be difficult, if not impossible, to analyse off-line all the possible combinations of processor loads. For this reason, it is worthwhile attempting to define new flexible architectures that enable computing systems to adapt to potential changes in the environment.

We assume a system composed by three main components: the first one is responsible of the management of the requests arisen when new tasks require to be executed. This management component asks to the second component about the resources available to accept the new tasks. The second component performs a feasibility analysis to determine if the new tasks can be accepted coping with its real-time constraints. A new processor speed is also computed. A third component monitors the execution of tasks applying a fixed priority scheduling policy and additionally controlling the frequency of the processor.

This paper focus on the second component providing a "correct" (a task never is accepted if it is not schedulable) and "near-exact" (a task is rarely rejected if it is schedulable) algorithm that can be applicable in practice because its low/medium and predictable computational cost. The algorithm analyses task admission in terms of processor frequency scaling. The paper presents the details of a novel algorithm to analyse tasks admission and processor frequency assignment. Additionally, we perform several simulations to evaluate the comparative performance of the proposed approach. This evaluation is made in terms of energy consumption, task rejection ratios, and real computing costs. The results of simulations show that from the cost, execution predictability, and task acceptance points of view, the proposed algorithm mostly outperforms other constant voltage scaling algorithms.

1. Introduction

Modern cyber-physical systems (CPS) assume a complex and dynamic interaction between the real world and the computing system in real-time (Wolf, 2009). In this scenario, computing systems are commonly formed of many embedded units that are heterogeneously networked. One of the challenges facing future systems is to address the separation of modal control design from the deployment of executable code. In such systems, it would be difficult, if not impossible, to analyze off-line all the possible combinations of processor loads. Because of the large range of control process domains it is difficult to address control modality only through the parameterization of pre-loaded local controllers. For this reason, it is worthwhile attempting to define new flexible architectures that enable computing systems to optimally adapt to potential changes in the environment in a dynamic manner.

From the point of view of embedded control systems, it is well known that the integration of controller design and real-time scheduling is necessary (Cervin, 2003). However, this approach is not enough in dynamic environments with limited computing resources. Therefore, additional mechanisms must be included in the design to optimize the use of resources and provide adaptation to the changing environmental conditions. Likewise, specific quality of services (QoS) levels must be guaranteed to ensure correct system operation. These QoS levels maintain a suitable control performance and a temporal predictability in the control system.

In this context, changes in the system operation mode, or in the environment, may force the disabling of some controllers and the enabling of others. In addition, controller switching mechanisms must be added in both local and distributed forms, resulting

* Corresponding author. Tel.: +34 665368512; fax: +34 963877579.
** Corresponding author. Tel.: +34 963877000x75706; fax: +34 963877579.
E-mail addresses: jacopa@ai2.upv.es (J.O. Coronel), jsimo@disca.upv.es (J.E. Simó).

0164-1212/$ – see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2011.11.284

in workload changes for the embedded and networked units (Emberson and Bate, 2007; Real and Crespo, 2004). This can have a significant impact on the overall performance and real-time constraints of the system. For this reason, task and component migration approaches (Briao et al., 2007) may contribute to on-line redistribution and reallocation of workload – according to each situation.

An example of this kind of architecture is proposed in Simarro et al. (2008), which is developed in the context of a distributed real-time control system and from the perspective of control kernel middleware (Crespo et al., 2006; Albertos et al., 2006). The proposal includes features such as control basis, controller switching, and code delegation. From the point of view of CPS, these approaches partially bridge the abstraction gap (Lee, 2006).

In summary, cyber-physical systems should be able to deliver new levels of performance and efficiency thanks to sophisticated control computing co-design. Control systems mean that we must change our understanding of computing systems and accept that cyber-physical systems actively engage with the real world and real-time restrictions – and so expend real energy.

This paper explores methods that can be used in task migration when combined with techniques of energy management. These methods can maximize overall energy savings; facilitate thermal chip management by moving tasks away from the hot processing unit; balance the workload or concentration of parallel processing elements; decrease communication among those tasks that are particularly energy-efficient on wireless sensor networks (WSN) systems (Yi et al., 2009; Yuan and Wang, 2008; Kumar and Manimaran, 2007); and help guarantee the fulfillment of real-time system constraints.

Several power-aware techniques have been widely addressed in real-time literature (Piao et al., 2009; Tavares et al., 2008; Aydin et al., 2006; Pillai and Shin, 2001). These papers have discussed dynamic voltage scaling (DVS) of the processor, also known as dynamic voltage and frequency scaling. This approach exploits the convex, and normally quadratic relationship between CPU energy consumption and voltage. Additionally, these techniques have been extended to reduce the energy consumed during memory cycles (Cho and Chang, 2006; Liang et al., 2008) and network energy consumption (Yi et al., 2009; Kumar et al., 2008).

Let's consider a dynamic environment where an embedded and networked system operates with the support of task migration and processor frequency scaling. Assuming that the system knows where and when it must allocate tasks (Briao et al., 2007; Emberson and Bate, 2007), we must perform feasibility analyses when each task arrives and departs. This guarantees that the temporal requirements of the system will be accomplished during the task allocation phase. Additionally, a new processor speed (frequency scaling) should be also computed to enable the system to adapt itself to the new computational workload and so reduce energy consumption. Although some authors have carried out these two phases (feasibility analysis and frequency scaling computation) separately (AlEnawy and Aydin, 2005), these analyses are strongly related and in some cases can be performed together.

The main motivation to use fixed priorities approaches is to accomplish the standard ARINC-653 (AASS Interface, 2003), which is used mainly in Avionics and currently also in Aerospace environment (Windsor and Hjortnaes, 2009). This standard defines a Time and Space partitioning system and each partition should be executed on fixed priority scheduling approaches. An example is the RTEMS O.S. on XtratuM hypervisor, which we are currently working (Galizzi et al., 2011, 2012). However, we use the EDF-based scheduler to compare with the proposed approach by suggestion of a reviewer. As is well known, the EDF algorithm has a higher performance than fixed priority based scheduling, but we have design constraints.

This work mainly focuses on a proposal for a new algorithm to be used on-line during the task allocation and processor speed assignment phases. The algorithm uses fixed priority scheduling schemes with deadlines less than, or equal to, the period of the tasks when the dynamic processor workloads are being handled.

1.1. Related work

Few works cover task migration in the context of embedded and networked systems. Moreover, most algorithms are aimed at multiprocessor systems with soft real-time constraints (Yazdi et al., 2008; Emberson and Bate, 2007; Anderson et al., 2005). Some papers have addressed both task migration and energy consumption minimization (Briao et al., 2007; Chen, 2005). Briao et al. (2007) analyze the impact of task migration and justify its use because it compensates for the performance and energy costs involved in the system.

Several heuristics has been proposed for task allocation problems with deadlines equal to the periods (Mejia-Alvarez et al., 2004; Chen, 2007) when earliest-deadline-first (EDF) scheduling is in use. AlEnawy and Aydin (2005) use algorithms for allocating real-time tasks by applying rate monotonic (RM) scheduling. In their article, a comparison of some admission control algorithms is presented in function of complexity, feasibility, and energy consumption parameters. However, only tasks with deadlines that are equal to the period are analyzed. Moreover, aspects such as predictability of execution, behaviour related to the arrival of tasks, and the real computing cost of the algorithms are not taken into account.

At the CPU-level, DVS techniques can be classified as *static* or *dynamic*. Static algorithms for hard real-time systems use parameters such as period or minimum inter-arrival time, and assume that each task executes its worst-case execution time when selecting the processor frequency, and that this is statically decided before execution. Dynamic algorithms are based on the reclamation of additional slack resulting from the early completions of tasks. These are then used to further reduce the processor frequency and save more energy. These algorithms are applied at run-time.

Using static algorithms we can obtain a single processor frequency that never changes, or obtain variable frequencies that are statically decided before execution. An example of the former is Saewong and Rajkumar (2003), in which an algorithm that chooses a single frequency for a fixed priority scheduling scheme is proposed. In Bini et al. (2005) the authors present a method for approximating any speed level with two given discrete values that are switched as a pulse width modulation signal in order to obtain the average value. For the latter, in Mejia-Alvarez et al. (2004) and Saewong and Rajkumar (2003), the authors propose an approach whereby each task is assigned a different frequency. Several authors have published works that find the optimal voltage schedule for recurrent tasks – examples for periodic tasks include Liu and Mok (2003) and Yun and Kim (2003) and examples for periodic and aperiodic tasks include Zhong et al. (2007) and Scordino and Lipari (2006). Furthermore, speed function and the characterization of the points in time at which speed changes occur has been presented in Liu and Mok (2003) and Gaujal and Navet (2007). However, a drawback in the works with statically established variable frequencies can appear if a task activation is lost or delayed. The drawback is that the entire frequency assignment will be affected, and this leads to missed deadlines.

Some authors, such as Piao et al. (2009), have proposed dynamic algorithms working in combination with static methods. These algorithms can be divided into inter-task and intra-task methods. In the inter-task algorithms, the processor frequency is determined task-by-task, whereas intra-task algorithms may adjust the frequency within the boundaries of a given task. In Kim et al. (2002), a performance comparison of several dynamic techniques is