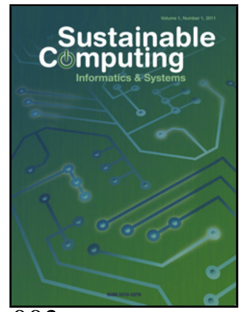
Accepted Manuscript



Title: Energy Aware Frame Based Fair Scheduling

Author: Sanjay Moulik Arnab Sarkar Hemangee K. Kapoor

PII: S2210-5379(17)30054-9
DOI: https://doi.org/doi:10.1016/j.suscom.2018.03.003
Reference: SUSCOM 238

To appear in:

Received date: 28-2-2017
Revised date: 7-12-2017
Accepted date: 27-3-2018

Please cite this article as: Sanjay Moulik, Arnab Sarkar, Hemangee K. Kapoor, Energy Aware Frame Based Fair Scheduling, *<![CDATA[Sustainable Computing: Informatics and Systems]]>* (2018), https://doi.org/10.1016/j.suscom.2018.03.003

Energy Aware Frame Based Fair Scheduling

Sanjay Moulik[1,*], Arnab Sarkar[1], Hemangee K. Kapoor[1]

Abstract

Energy has now-a-days become a critical resource in all battery operated devices. Reduction of energy consumption is essential to prolong the battery life in these systems. A significant class of such systems often execute a mix of independent real-time applications with guaranteed Quality of Service (QoS) requirements. Proportional fair schedulers form a very effective mechanism for handling such application sets, with high resource utilisation on multiprocessor systems. In this paper, we propose a DVFS enabled proportional fair strategy called *Energy Aware Frame Based Fair Scheduling (EAFBFS)* for executing a set of real-time QoS aware tasks on multicores. In addition to providing appreciable energy savings, EAFBFS is able to deliver high and tunable fairness accuracy, low scheduling related overheads along with resource utilisation optimality, by combining and leveraging the benefits of two prominent state-of-the-art schedulers ERFair and DPFair. While the proposed scheduler ensures atmost m-1 task migrations over stipulated intervals called *time frames* similar to DPFair, EAFBFS exhibits far improved fairness properties. Experimental results show that EAFBFS performs almost at par with energy aware DPFair in terms of energy savings and migrations incurred while achieving much higher proportional fairness accuracy (10-15 times on average over DPFair).

Keywords: EAFBFS, Energy Aware, ERFair, DPFair, Frame, Real-Time and Scheduler

1. Introduction

Many real-time multiprocessor embedded systems execute their applications in a proportional fair manner. Proportional fair or rate based execution progress guarantees are typically of the form, *complete X units of execution of application A out of every Y time units.* Such fairness guarantees are often useful in executing a mix of applications with various degrees of timeliness criticality such as online gaming, streaming audio and video, email, web browsing etc. [1]. A well known soft real-time proportional fair scheduling strategy is the *Completely Fair Scheduler* (CFS) [2]. CFS is the current default scheduler in the Linux kernel and is at the heart of all Android enabled embedded devices. On the other hand, the ERFair scheduler [3] is an important hard real-time proportional fair algorithm. ERFair scheduling is a work-conserving version[2] of the famous Pfair algorithm(*PF* [4, 5], *PD* [6], PD^2 [7]), which is the first known optimal real-time multiprocessor scheduler.

With advancement of technology, a newer classes of devices like mobiles, laptops, PDAs etc. have emerged, which depend upon battery as their primary source of energy. Being battery operated, efficient energy usage has become a critical design constraint in all these devices. Hence, lot of research has been conducted towards their power management at various levels of abstraction, starting from hardware and firmware to architectural, system and even application levels. At the operating system level, two primary mechanisms are generally used to reduce energy consumption: 1. Dynamic Voltage and Frequency Scaling (DVFS) [8–10] and 2. Dynamic Power Management (DPM) [8, 11, 12]. DVFS involves dynamic adaptation of a processor's operating voltage/frequency according to instantaneous requirements of the workload being handled at a given time. As energy dissi-

*Corresponding author
Email address: s.moulik@iitg.ac.in (Sanjay Moulik)
[1]All the authors are with Department of Computer Science and Engineering, Indian Institute of Technology, Guwahati - 781039, Assam,India.

[2]A work conserving scheduler never allows a processor to sit idle when there are ready tasks available to run