Journal of King Saud University – Computer and Information Sciences (2017) 29, 74–84

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

Executed-time Round Robin: EtRR an online non-clairvoyant scheduling on speed bounded processor with energy management

Pawan Singh *, **Berhane Wolde-Gabriel**

School of Informatics, Institute of Technology, Hawassa University, Ethiopia

Received 20 July 2015; revised 11 March 2016; accepted 17 March 2016
Available online 31 March 2016

KEYWORDS

Weighted flow time;
Power Management;
Non-clairvoyant scheduling;
Online scheduling;
Potential analysis

Abstract Energy conservation has become a prime objective due to excess use and huge demand of energy in data centers. One solution is to use efficient job scheduling algorithms. The scheduler has to maintain the machine's state balance to obtain efficient job schedule and avoid unnecessary energy consumption. Although the practical importance of non-clairvoyant scheduling problem is higher than clairvoyant scheduling, in the past few years the non-clairvoyant scheduling problem has been studied lesser than clairvoyant scheduling. In this paper, an online non-clairvoyant scheduling problem is studied to minimize total weighted flow time plus energy and a scheduling algorithm Executed-time Round Robin (EtRR) is proposed. Generally, weights of jobs are system generated and they are assigned to jobs at release/arrival time. In EtRR, the weights are not generated by the system, rather by the scheduler using the executed time of jobs. EtRR is a coupling of weighted generalization of Power Management and Weighted Round Robin (WRR). We adopt the conventional power function $P = s^{\alpha}$, where s and $\alpha > 1$ are speed of a processor and a constant, respectively. EtRR is $O(1)$-competitive, it is using a processor with the maximum speed $(1 + \tau/3)T$, where the maximum speed of optimal offline adversary is T and $0 < \tau \leqslant (3\alpha)^{-1}$.
© 2016 The Authors. Production and hosting by Elsevier B.V. on behalf of King Saud University. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

1. Introduction

"What matters most to the computer designers at Google is not speed, but power, low power, because data centers can consume as much energy as a city." (Markoff and Lohr, 2002). In the current epoch, energy conservation is a key issue in designing modern processors. Dynamic speed scaling is adapted by many chip manufacturers and they produce associated software also such as AMD's PowerNow. These softwares ease an operating system to scale the processor's speed and obtain energy efficiency. Modern scheduling

* Corresponding author.
E-mail addresses: pawansingh3@yahoo.com, pawansingh51279@gmail.com, dr_pawansingh@hu.edu.et (P. Singh), berhane.woldegabriel@gmail.com (B. Wolde-Gabriel).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2016.03.002
1319-1578 © 2016 The Authors. Production and hosting by Elsevier B.V. on behalf of King Saud University.
This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

algorithms comprise of two components: first, a job selection policy that determines which job to execute; second, a speed scaling policy that computes the execution speed of a processor, at any time.

An operating system has dual conflicting objectives to solve such problems: first, to optimize some scheduling Quality of Service (QoS) objective; second, Power Management (PM) objective, such as total weighted flow time and total energy used, respectively (Bansal et al., 2009). Scheduling jobs becomes complicated, if QoS, speed scaling and energy usage efficiency are considered at once. A scheduler arranges jobs in some order to optimize a certain QoS metric, such as throughput, makespan, slowdown, flow time or weighted flow time. In most of the operating systems (such as UNIX), when job arrives there is no information about the job's size. In clairvoyant (non-clairvoyant) scheduling algorithms the sizes of jobs are known (not known) at the release time. Unlike online, offline algorithms know complete job sequence in advance, which is not possible in most of the practical problems. Yao et al. (1995) initiated the theoretical study of speed scaling and proposed a model, wherein the processor's speed s can vary from zero to infinity, i.e., $[0, \infty)$. The traditional power consumption function is $P = s^{\alpha}$, where $\alpha > 1$ a constant, s speed of a processor and P is the power consumed (the value of $\alpha = 2$ or 3 for CMOS-based chips (Pruhs et al., 2008)). There are two speed models, unbounded speed and bounded speed model, where the speed ranges are $[0, \infty)$ and $[0, T]$, respectively (Bansal et al., 2009). Kalyanasundaram and Pruhs (2000) introduced an idea to augment the resources of the non-clairvoyant scheduler by increasing the processor's speed. As per Kalyanasundaram and Pruhs (2000), if a non-clairvoyant scheduler is allowed $(1 + \tau)$ times faster processor, then it can attain a response time within a $(1 + 1/\tau)$ factor achievable by the best possible clairvoyant algorithm.

Motwani et al. (1994) first analyzed non-clairvoyant scheduling algorithm for the objective of mean response time and showed that Round Robin (RR) has a performance ratio of $\left(2 - \frac{2}{(n+1)}\right)$, which is optimal for deterministic non-clairvoyant algorithms; they proved that the lower bounds remain equal for the jobs of bounded sizes, i.e. the ratio of the largest to the smallest execution time is bounded by some small constant. RR makes $\Omega(x)$ preemptions for a job of size x. The randomized algorithms have the same performance ratio as RR. Any deterministic non-clairvoyant dynamic algorithm has performance ratio $\Omega(n^{1/3})$, while any randomized non-clairvoyant dynamic algorithm has a performance ratio $\Omega(\log n)$. Muthukrishnan et al. (1999) studied uniprocessor online job scheduling algorithm with slowdown or stretch as their objective and showed that SRPT is 2-competitive but in clairvoyant settings. Berman and Coulston (1999) considered the problem of online preemptive non-clairvoyant scheduling (Balance) on a uniprocessor model for the objective of minimizing the total response time. Balance schedules the least processed job first. Berman and Coulston (1999) proved that if the Balance runs υ times faster than the clairvoyant algorithm then the competitive ratio is $(\upsilon/(\upsilon - 1))$ at most and for $\upsilon \geqslant 2$ the competitive ratio of Balance is $(2/\upsilon)$; they concluded that adequately high speed is more powerful than clairvoyance. Edmonds (2000) achieved $\Omega(\sqrt{n})$ lower bound on competitive ratio of sequential and parallelizable jobs for randomized non-clairvoyant schedulers; if the speed of processor is $(1 + \tau)$, then the lower bound is $\Omega\left(\frac{1}{\tau}\right)$. Edmonds (2000) proved that after the resource augmentation, when speed of a processor $s > 2$, the Equi-partition and Processor Scheduling (Round Robin), which shares the processor equally among all jobs, becomes competitive. In case, if there are p processors of $(2 + \tau)$ speed, then the competitive ratio of Equi-partition is between $\frac{2}{3}\left(1 + \frac{1}{\tau}\right)$ and $\left(2 + \frac{4}{\tau}\right)$; with extra augmentation, when $s \geqslant 4$ the competitive ratio of Equi-partition is between $\left(\frac{2}{s}\right)$ and $\left(\frac{16}{s}\right)$.

As per Kalyanasundaram and Pruhs (2003) and Becchetti and Leonardi (2004), randomized version of Multi Level Feedback Queue algorithm is $O(logn)$-competitive. Yun and Kim (2003) proposed that it is NP-hard to calculate a minimum energy schedule for jobs with fixed priority. Becchetti et al. (2006) showed the modification in Bansal's algorithmic result and gave $O(\alpha^2/log^2\alpha)$ competitive algorithm with resource augmentation for the objective of minimizing weighted flow time plus energy. Bansal et al. (2007) showed that the algorithm Optimal Available (OA) is -competitive using the potential analysis and the competitive ratio is $\mu_{\tau}\gamma$, where $\gamma = \max\left\{2, \frac{2(\alpha-1)}{\alpha-(\alpha-1)^{1-1/\alpha-1}}\right\}$ and $\mu_{\tau} = \max\{(1 + 1/\tau),\ (1 + \tau)\alpha\}$ for any $\tau > 0$. Bansal et al. (2009) assumed that allowable speeds are countable collection of disjoint subintervals in range $[0, \infty)$ and they have taken the power functions that are non-negative, continuous and differentiable. Bansal et al. (2009) used SRPT for job selection and the speed scaling such that at any time the speed is equal to one plus number of unfinished jobs, their algorithm is $(3 + \tau)$-competitive for the objective of total flow time plus energy. Bansal et al. (2009) considered Highest Density First (HDF) also for job selection and the speed scaling such that at any time the speed is equal to fractional weigh of unfinished jobs, and gave a $(2 + \tau)$-competitive algorithm for the objective of fractional weighted flow time plus energy.

In multiprocessor systems, a new concept of sleep management, QoS and energy consumption were used by Albers (2010). The non-clairvoyant speed scaling scheduling algorithm LAPS proposed by Gupta et al. (2012) is $(1 + \tau)$-speed, $O(1/\tau^5)$-competitive for the objective of minimizing the flow time plus energy on related machines. Gupta et al. (2012) gave the first scalable non-clairvoyant algorithm for speed-scalable heterogeneous processors for fixed-speed related machines and suggested that scheduling heterogeneous multiprocessors might be inherently more complex than scheduling homogeneous multiprocessors, or at least, require significantly unlike algorithms. Chan et al. (2013) gave an online non-clairvoyant deterministic algorithm Scheduling with Arrival Time Alignment (SATA) with sleep management, which is $(1 + \tau)$-speed, $O(1/\tau^2)$-competitive for the objective of minimizing total flow time plus energy. SATA uses mechanism called the arrival-time-alignment to ensure the even jobs distribution when a job arrives or finishes, wherein it migrates each job at most four times on an average. In classical settings with no sleep management SATA is $(1 + \tau)$-speed, $8(1 + 1/\tau)^2$-competitive for the objective to minimize flow time only.

Fox et al. (2013) proposed a non-clairvoyant algorithm Weighted Latest Arrival Processor Sharing with Energy (WLAPS + E), which is $(1 + 6\tau)$-speed $(5/\tau^2)$-competitive, where $0 < \tau \leqslant 1/6$, for the objective of weighted flow time plus energy. WLAPS + E schedules late arriving jobs and a job can

Download English Version:

https://daneshyari.com/en/article/4960337

Download Persian Version:

https://daneshyari.com/article/4960337

Daneshyari.com