ARTICLE IN PRESS YINCO:4327

Information and Computation ••• (••••) •••–•••

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Scheduling on power-heterogeneous processors ☆

Susanne Albers [a,1], Evripidis Bampis [b,2], Dimitrios Letsios [c,*,3], Giorgio Lucarelli [d,4], Richard Stotz [a]

[a] Fakultät für Informatik, Technische Universität München, Germany
[b] Sorbonne Universités, UPMC Univ. Paris 06, UMR 7606, LIP6, Paris, France
[c] Université Côte d'Azur, Inria, CNRS, I3S, Sophia Antipolis, France
[d] Université Grenoble-Alpes, INP, UMR 5217, LIG, France

ARTICLE INFO

Article history:
Received 19 March 2016
Received in revised form 10 August 2017
Available online xxxx

Keywords:
Heterogeneous processors
Scheduling
Speed-scaling
Energy
Approximation algorithms
Online algorithms

ABSTRACT

We consider the problem of scheduling a set of jobs, each one specified by its release date, its deadline and its processing volume, on a set of heterogeneous speed-scalable processors, where the energy-consumption rate is processor-dependent. Our objective is to minimize the total energy consumption when both the preemption and the migration of jobs are allowed. We propose a new algorithm based on a compact linear programming formulation. Our method approaches the value of the optimal solution within any desired accuracy for a large set of continuous power functions. Furthermore, we develop a faster combinatorial algorithm based on flows for standard power functions and jobs whose density is lower bounded by a small constant. Finally, we extend and analyze the AVerage Rate (AVR) online algorithm in the heterogeneous setting.

1. Introduction

Nowadays energy consumption of computing devices is an important issue in both industry and academia. One of the main technological alternatives in order to take into account the energy consumed in modern computer systems is based on the use of speed-scalable processors where the speed of a processor may be dynamically changed over time. When a processor runs at speed s, then the rate with which the energy is consumed (i.e., the power) is $f(s)$ with f being a non-decreasing function of the speed. According to the well-known cube-root rule for CMOS devices, the speed of a device is proportional to the cube-root of the power and hence $f(s) = s^3$. However, the standard model that is usually studied in the literature considers that the power is $f(s) = s^\alpha$ with $\alpha > 1$ a constant. Other works consider that the power is an arbitrary convex function [7,9].

The algorithmic study of this area started with the seminal paper of Yao et al. [20], where a set of jobs, each one specified by its work, its release date and its deadline, has to be scheduled preemptively on a single processor so that the

☆ A preliminary version has been presented to LATIN 2016.

* Corresponding author.
E-mail addresses: albers@in.tum.de (S. Albers), Evripidis.Bampis@lip6.fr (E. Bampis), dletsios@unice.fr (D. Letsios), giorgio.lucarelli@inria.fr (G. Lucarelli), stotz@in.tum.de (R. Stotz).

[1] Work supported by the European Research Council, Grant Agreement No. 691672, and the German Research Foundation, project AL464/9-1.
[2] Work supported by project GDR-RO AGaPe of CNRS.
[3] Work supported by ANR project Stint and ANR program “Investments for the Future”.
[4] Work supported by ANR project Moebus.

https://doi.org/10.1016/j.ic.2017.09.013
0890-5401/© 2017 Published by Elsevier Inc.

energy consumption is minimized. In [20], an optimal polynomial-time algorithm has been proposed for this problem, while Li et al. [19] proposed an optimal algorithm with lower running time. The *homogeneous multiprocessor* setting in which the *preemption* and the *migration* of the jobs are allowed has been also studied. Chen et al. [12] proposed a greedy algorithm if all jobs have common release dates and deadlines. Bingham and Greenstreet [11] presented a polynomial-time algorithm for the more general problem with arbitrary release dates and deadlines. Their algorithm is based on solving a series of linear programs. Since the complexity of this algorithm can be high for practical applications, Albers et al. [1] and Angel et al. [3], independently, have been interested in the design of a combinatorial algorithm. Both works are based on the computation of several maximum flows in appropriate networks.

Albers et al. [1] have also considered the online version of the multiprocessor problem with homogeneous processors and they studied two well-known algorithms, namely the *Optimal Available (OA)* and the *Average Rate (AVR)*, which have been originally proposed by Yao et al. in [20] for the single-processor setting. Specifically, they proved that OA is α^α-competitive and that AVR is $(2^{\alpha-1}\alpha^\alpha + 1)$-competitive. Note that, for the single-processor case, the competitive ratio of OA cannot be better than α^α [10], while the lower bound for AVR is $2^{\alpha-1}\alpha^\alpha$ [8].

When migrations of jobs are not allowed, the problem is strongly $\mathcal{NP}$-hard even in the special case with homogeneous processors and unit work jobs [2]. Furthermore, there exists a $(1+\epsilon)^\alpha \widetilde{B}_\alpha$-approximation algorithm, where α is the maximum power exponent and $\widetilde{B}_\alpha$ is the α-generalized Bell number [6]. When both migrations and preemptions are forbidden, the problem is strongly $\mathcal{NP}$-hard even in the case with a single processor [4]. In this setting, the approximability of the problem is an open question. For the special case with homogeneous processors, there exists a $O((w_{\max}/w_{\min})^\alpha)$-approximation algorithm [5] as well as a quasi-polynomial time approximation scheme (QPTAS) producing a $(1+\epsilon)$-approximate solution in $n^{O(\text{polylog}(n))}$ time [18].

In this paper, we consider the problem of scheduling a set of jobs on a set of *power-heterogeneous* processors when the preemption and the migration of the jobs are allowed. In our setting, each processor P_p is characterized by its own power function. This means that if a processor P_p runs at speed s, then its power is given by a non-decreasing function $f_p(s)$. The motivation to study power-aware scheduling problems is based on the need for more efficient computing. Indeed, parallel heterogeneous systems with multiple cores running at lower frequencies offer better performances than a single core. However, in order to exploit the opportunities offered by the heterogeneous systems, it is essential to focus on the design of new efficient power-aware algorithms taking into account the heterogeneity of these architectures. In this direction, some recent papers [6,16,17] have studied the impact of the introduction of the heterogeneity on the difficulty of some power-aware scheduling problems. Especially in [16], Gupta et al. show that well-known priority scheduling algorithms that are energy-efficient for homogeneous systems become energy inefficient for heterogeneous systems.

For the case where job migrations are allowed and the heterogeneous power functions are convex, an algorithm has been proposed in [6] that returns a solution within an additive factor of ϵ far from the optimal and runs in time polynomial to the size of the instance and to $1/\epsilon$. This result generalizes the results of [1,3,7,11] from the homogeneous setting to the heterogeneous one. However, the algorithm proposed in [6] is based on solving a configuration linear program using the Ellipsoid method. Given that this method may not be very efficient in practice, we focus on other approaches. We first propose a polynomial-time algorithm based on a compact linear programming formulation which solves the problem within any desired accuracy. Our algorithm does not need the use of the Ellipsoid method like in [6] and it applies for a large family of continuous non-decreasing power functions.

The above result leaves open a natural question: *is it possible to generalize the flow-based approach used in [1,3] for the homogeneous multiprocessor problem to the power-heterogeneous case?* This question is interesting even for standard power functions of the form $f_p(s) = s^{\alpha_p}$. This last case is the goal of the second part of our paper. However, when power-heterogeneous processors are considered some structural properties of the optimal schedules of the homogeneous case are no longer valid. For instance, in the heterogeneous setting, in any optimal schedule, the speed of a job is not necessarily unique, but it may change when parts of the same job are executed on different processors. A second difficulty comes from the fact that, while in the homogeneous case the processor on which a job is executed at a given time has no influence on the energy consumption, this is a crucial decision when scheduling on heterogeneous multiprocessors. Here, we overcome these subtle difficulties and propose a max-flow based algorithm which is more complicated than its homogeneous counterpart (for example, the network formulation is more enhanced). In particular, we show that it produces a solution arbitrarily close to the optimal for jobs whose density is lower bounded by a small constant; this constant depends on the exponents of the power functions. The above assumption ensures that no job is processed with a speed less than one by any processor and allows us to solve the problem by performing maximum flow computations in a principled way. Note that this assumption is reasonable in practice because the speed of a processor is multiple CPU cycles per second.

The third part of our paper is devoted to the analysis of the well known online algorithm AVR. Our analysis simplifies the analysis in [1] for the homogeneous case and allows us to extend it in the power-heterogeneous setting. Specifically, we prove that Heterogeneous-AVR is $((1+\epsilon)(\rho+1))$-competitive algorithm for arbitrary power functions, where ρ is the worst competitive ratio of the single-processor AVR algorithm among all processors. This turns to be $((1+\epsilon)(\alpha^\alpha 2^{\alpha-1}+1))$-competitive algorithm for standard power functions of the form $f_p(s) = s^{\alpha_p}$, where α is the maximum power exponent among all processors.

In the following section we formally define our problem and we give the notation that we use. In Section 3, we present our LP-based algorithm, while in Section 4 we describe a flow-based combinatorial algorithm. Finally, the Heterogeneous-AVR and its analysis are given in Section 5.

Please cite this article in press as: S. Albers et al., Scheduling on power-heterogeneous processors, Inf. Comput. (2017), https://doi.org/10.1016/j.ic.2017.09.013