Energy Efficient Multiprocessor Scheduling via Configuration LP

Evripidis Bampis* Alexander Kononov† Dimitrios Letsios*‡
Giorgio Lucarelli*‡ Maxim Sviridenko§

1 Introduction

Energy minimization is an important issue in our days [8]. One of the main mechanisms for reducing the energy consumption in modern computer systems is the use of speed scalable processors. Starting from the seminal paper of Yao et al. [16], many papers adopted the speed-scaling model in which if a processor runs at speed s, then the rate of the energy consumption, i.e., the power, is $P(s) = s^\alpha$ with α a constant. This model captures the intuitive idea that the faster a processor works the more energy it consumes. Recently, a series of papers [12, 11, 10] have proposed an extension of this model in order to capture the heterogeneity of the forthcoming computer architectures [6, 13, 14, 15]. In their model, each processor has its own power function. This means that a job executed with the same speed on different machines consumes more or less energy.

Our work is in the same direction. We consider a *fully heterogeneous environment* where both, the jobs' characteristics are processor-dependent and every processor has its own power function. Formally, we consider the following problem: we are given a set $\mathcal{J}$ of n jobs and a set $\mathcal{P}$ of m parallel processors. Every processor $i \in \mathcal{P}$ obeys to a different speed-to-power function, i.e., it is associated with a different $\alpha_i > 2$ and hence if a job runs at speed s on processor i, then the power is $P(s) = s^{\alpha_i}$. Each job $j \in \mathcal{J}$ has a different release date $r_{i,j}$, deadline $d_{i,j}$ and work $w_{i,j}$ in each processor $i \in \mathcal{P}$. The assumption that the jobs have processor-dependent works is well suited for computer systems composed by a set of specialized processors, each one being able at executing efficiently a particular type of jobs. In such a system different works for the same job may correspond to different level of quality. Moreover, processor-dependent release dates are interesting for the non-migratory variant when the processors are connected by a network. In such a case we may assume that every job is initially available at a given processor and that a transfer time must elapse before it becomes available at a new machine [4, 7].

We consider two variants of the multiprocessor *preemptive* problem: the migratory and the non-migratory. In both cases, the execution of a job may be interrupted and resumed later. In the *migratory* case each job may be executed by more than one

*{Evripidis.Bampis, Giorgio.Lucarelli}@lip6.fr. LIP6, Univ. Pierre et Marie Curie, France.
†alvenko@math.nsc.ru. Sobolev Institute of Mathematics, Novosibirsk, Russia.
‡dimitris.letsios@ibisc.univ-evry.fr. IBISC, Univ. d' Évry, France.
§M.I.Sviridenko@warwick.ac.uk. Department of Computer Science, University of Warwick, UK.

processors, without allowing parallel execution of a job. If x units of work for the job j are executed on the processor i, then $\frac{x}{w_{i,j}}$ portion of j is accomplished by i. In the *non-migratory* case each job has to be entirely executed on a single processor. In the last part of the paper we focus on the *non-preemptive* single-processor case.

2 Related work

Yao et al. [16] proposed an optimal algorithm for finding a feasible preemptive schedule with minimum energy consumption when a single-processor is available. The multiprocessor case has been solved optimally in polynomial time when both the preemption and the migration of jobs are allowed [1, 5]. Albers et al. [2] considered the multiprocessor problem, where the preemption of the jobs is allowed, but not their migration. They proved that the problem is $\mathcal{NP}$-hard even for instances with common release dates and common deadlines. Greiner et al. [9] gave a generic reduction transforming an optimal schedule for the multiprocessor problem with migration, to a B_α-approximate solution for the multiprocessor problem with preemptions but without migration, where B_α is the α-th Bell number. Antoniadis and Huang [3] proved that the single-processor non-preemptive problem is $\mathcal{NP}$-hard and they proposed a $2^{5\alpha-4}$-approximation algorithm. All above multiprocessor results concern the *homogeneous* case.

3 Results

We consider first the heterogeneous multiprocessor speed-scaling problem when the preemption and the migration of jobs are allowed. We formulate the problem as a configuration LP with an exponential number of variables and a polynomial number of constraints. To deal with this, we consider the dual LP and we show how to apply the ellipsoid algorithm to it. As we perform a geometric rounding in order to discretize the possible speed values, our solution is an additive factor ε far from the optimal.

Theorem 1 *A schedule for the heterogeneous multiprocessor speed-scaling problem with migrations of energy consumption $OPT+\varepsilon$ can be found in polynomial time with respect to the size of the instance and to $1/\log(1+\varepsilon)$.*

Then, we pass to the heterogeneous multiprocessor preemptive speed-scaling problem where the migration of jobs is not permitted, which is known to be $\mathcal{NP}$-hard [2]. We formulate the problem as a configuration IP and we follow the same approach as before in order to solve the LP relaxation. Then, we perform a randomized rounding and we get the approximation result of the following theorem. Note that, it is possible to derandomize our algorithm using the method of conditional expectations.

Theorem 2 *In the case where $\alpha_{\max} = \max_{i\in\mathcal{P}}\{\alpha_i\} \geq 2$, there is an approximation algorithm which achieves a ratio of $(1+\varepsilon)B_{\alpha_{\max}}$ for the heterogeneous multiprocessor speed-scaling problem without migrations in time polynomial to n and to $1/\varepsilon$.*

Finally, we exploit this last result in order to improve the approximation ratio for the single-processor non-preemptive speed-scaling problem. We propose an approximation algorithm that gives better approximation guarantees for any $\alpha < 114$ with respect to the best known algorithm of Antoniadis and Huang [3]. Note that in practical applications α is usually between two and three.

Theorem 3 *The single-processor speed-scaling problem without preemptions can be approximated within a factor of* $2^{\alpha-1}(1+\varepsilon)B_\alpha$.

References

[1] S. Albers, A. Antoniadis, and G. Greiner. On multi-processor speed scaling with migration: extended abstract. In *SPAA'11*, pages 279–288. ACM, 2011.

[2] S. Albers, F. Müller, and S. Schmelzer. Speed scaling on parallel processors. In *SPAA'07*, pages 289–298. ACM, 2007.

[3] A. Antoniadis and C.-C. Huang. Non-preemptive speed scaling. In *SWAT'12*, volume 7357 of *LNCS*, pages 249–260. Springer, 2012.

[4] B. Awerbuch, S. Kutten, and D. Peleg. Competitive distributed job scheduling (Extended abstract). In *STOC'92*, pages 571–580, 1992.

[5] E. Bampis, D. Letsios, and G. Lucarelli. Green scheduling, flows and matchings. In *ISAAC'12*, volume 7676 of *LNCS*, pages 106–115. Springer, 2012.

[6] F. A. Bower, D. J. Sorin, and L. P. Cox. The impact of dynamically heterogeneous multicore processors on thread scheduling. *IEEE Micro*, 28:17–25, 2008.

[7] X. Deng, H.-N. Liu, and B. Xiao. Deterministic load balancing in computer networks. In *SPDP'90*, pages 50–57, 1990.

[8] J. Glanz. Power, pollution and the internet, September 22, 2012. The New York Times.

[9] G. Greiner, T. Nonner, and A. Souza. The bell is ringing in speed-scaled multiprocessor scheduling. In *SPAA'09*, pages 11–18. ACM, 2009.

[10] A. Gupta, S. Im, R. Krishnaswamy, B. Moseley, and K. Pruhs. Scheduling heterogeneous processors isn't as easy as you think. In *SODA'12*, pages 1242–1253, 2012.

[11] A. Gupta, R. Krishnaswamy, and K. Pruhs. Nonclairvoyantly scheduling power-heterogeneous processors. In *Green Computing Conference*, pages 165–173, 2010.

[12] A. Gupta, R. Krishnaswamy, and K. Pruhs. Scalably scheduling power-heterogeneous processors. In *ICALP'10 (1)*, pages 312–323, 2010.

[13] R. Kumar, D. M. Tullsen, P. Ranganathan, N. P. Jouppi, and K. I. Farkas. Single-ISA heterogeneous multi-core architectures for multithreaded workload performance. In *ISCA'04*, pages 64–75, 2004.

[14] R. Merritt. Cpu designers debate multi-core future, February 2008. EE Times.

[15] T. Y. Morad, U. C. Weiser, A. Kolodnyt, M. Valero, and E. Ayguadé. Performance, power efficiency and scalability of asymmetric cluster chip multiprocessors. *Computer Architecture Letters*, 5:14–17, 2006.

[16] F. Yao, A. Demers, and S. Shenker. A scheduling model for reduced CPU energy. In *FOCS'95*, pages 374–382, 1995.