Energy Efficient Scheduling and Routing via Randomized Rounding

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

Abstract

We propose a unifying framework based on configuration linear programs and randomized rounding, for different energy optimization problems in the dynamic speed-scaling setting. We apply our framework to various scheduling and routing problems in heterogeneous computing and networking environments. We first consider the energy minimization problem of scheduling a set of jobs on a set of parallel speed-scalable processors in a fully heterogeneous setting. For both the preemptive-non-migratory and the preemptive-migratory variants, our approach allows us to obtain solutions of almost the same quality as for the homogeneous environment. By exploiting the result for the preemptive-non-migratory variant, we are able to improve the best known approximation ratio for the single processor non-preemptive problem. Furthermore, we show that our approach allows to obtain a constant-factor approximation algorithm for the power-aware preemptive job shop scheduling problem. Finally, we consider the min-power routing problem where we are given a network modeled by an undirected graph and a set of uniform demands that have to be routed on integral routes from their sources to their destinations so that the energy consumption is minimized. We improve the best known approximation ratio for this problem.

1 Introduction

We focus on energy minimization problems in heterogeneous computing and networking environments in the dynamic speed-scaling setting. For many years, the exponential increase of processors' frequencies followed Moore's law. This is no more possible because of physical (thermal) constraints. Today, for improving the performance of modern computing systems, designers use parallelism, i.e., multiple cores running at lower frequencies but offering better performances than a single core. These systems can be either homogeneous where an identical core is used many times, or heterogeneous combining general-purpose and special-purpose cores. Heterogeneity offers the possibility of further improving the performance of the system by executing each job on the most appropriate type of processors [6]. 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, Gupta et al. in [8] have studied the impact of the heterogeneity on the difficulty of various power-aware scheduling problems.

We show that rounding configuration linear programs helps in handling the heterogeneity of both the jobs and the processors. We adopt one of the main mechanisms for reducing the energy consumption in modern computer systems which is based on the use of speed scalable processors. Starting from the seminal paper of Yao et al. [10], many papers adopted the speed-scaling model in which if a processor

[*]{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.

runs at speed s, then the rate of the energy consumption, i.e., the power, is $P(s) = s^{\alpha}$ with α a constant close to 3 (new studies show that α is rather smaller: 1.11 for Intel PXA 270, 1.62 for Pentium M770 and 1.66 for a TCP offload engine [9]). Moreover, the energy consumption is the integral of the power over time. This model captures the intuitive idea that the faster a processor runs the more energy it consumes.

We first 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 $\alpha_i \geq 1$ and hence if a job runs at speed s on processor i, then the power is $P(s) = s^{\alpha_i}$. Let $\alpha = \max_i\{\alpha_i\}$. Each job $j \in \mathcal{J}$ has a different release date $r_{i,j}$, deadline $d_{i,j}$ and workload $w_{i,j}$ if job j is executed on processor $i \in \mathcal{P}$. The goal is to find a schedule of minimum energy respecting the release dates and the deadlines of the jobs.

In this paper we propose a unifying framework for minimizing energy in different heterogeneous computing and networking environments. We first consider two variants of the heterogeneous multiprocessor *preemptive* problem. In both cases, the execution of a job may be interrupted and resumed later. In the *non-migratory* case each job has to be entirely executed on a single processor. In the *migratory* case each job may be executed by more than one processors, without allowing parallel execution of a job. We also focus on the *non-preemptive* single processor case. Furthermore, we consider the energy minimization problem in an heterogeneous *job shop* environment where the jobs can be preempted. Finally, we consider the *min-power routing problem*, introduced in [3], where a set of uniform demands have to be routed on integral routes from their sources to their destinations so that the energy consumption to be minimized. We believe that our general techniques will find further applications in energy optimization.

2 Related Work and Our Contribution

Yao et al. [10] 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. [7] 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 [4] 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.

In this work we formulate heterogeneous scheduling and routing problems using *configuration linear programs (LPs)* and we apply *randomized rounding*. In Section 3, we consider the heterogeneous multiprocessor speed-scaling problem without migrations and we propose an approximation algorithm of ratio $(1+\varepsilon)\tilde{B}_{\alpha}$, where $\tilde{B}_{\alpha}$ corresponds to the α-th fractional moment of the Poisson distribution (generalized Bell number). For real values of α our result improves the $B_{\lceil \alpha \rceil}$ approximation ratio of [7] for the homogeneous case to $(1+\varepsilon)\tilde{B}_{\alpha}$ for the fully heterogeneous environment that we consider here (see Table 1). Our work includes some additional results which are omitted due to space constraints. To begin with, we propose an algorithm for the heterogeneous multiprocessor speed-scaling problem with migration. This algorithm returns a solution within an additive factor of ε far from the optimal solution and runs in time polynomial to the size of the instance and to $1/\varepsilon$. This result generalizes the results of [1, 5] from an homogeneous environment to a fully heterogeneous environment. Moreover, we transform the single processor speed-scaling problem without preemptions to the heterogeneous multiprocessor problem without migrations and we give an approximation algorithm of ratio $2^{\alpha-1}(1+\varepsilon)\tilde{B}_{\alpha}$, improving upon the previous known $2^{5\alpha-4}$-approximation algorithm in [4] for any $\alpha < 114$ (see Table 1). We also address the power-aware preemptive job shop scheduling problem and we propose a $((1+\varepsilon)\tilde{B}_{\alpha})$-

approximation algorithm, where μ is the number of all the operations. Finally we improve the analysis for the min-power routing problem with uniform demands given in [3], based on the randomized rounding analysis. Our approach gives an approximation ratio of $\tilde{B}_\alpha$ significantly improving the analysis given in [3] (see Table 1).

Value of α	Preemptive without migrations		Non-preemptive single processor		Routing uniform demands	
	Homogeneous [7]	Heterogeneous This paper	[4]	This paper	[3]	This paper
1.11	2	1.07(1+ε)	2.93	1.15(1+ε)	375	1.07
1.62	2	1.49(1+ε)	17.15	2.30(1+ε)	2196	1.49
1.66	2	1.54(1+ε)	19.70	2.43(1+ε)	2522	1.54
2	2	2(1+ε)	64	4(1+ε)	8193	2
2.5	5	3.08(1+ε)	362	8.72(1+ε)	46342	3.08
3	5	5(1+ε)	2048	20(1+ε)	262145	5

Table 1: Comparison of our approximation ratios vs. better previous known ratios for: (i) the preemptive multiprocessor problem without migrations, (ii) the single processor non-preemptive problem, and (iii) the min-power routing problem.

3 Heterogeneous Multiprocessor without Migrations

In this section, we present an approximation algorithm for fully heterogeneous environments when migrations of jobs are not allowed but preemptions are permitted.

In order to formulate the problem as a configuration integer program (IP) we discretize the time as follows. For each job $j \in \mathcal{J}$ and processor $i \in \mathcal{P}$, we partition the interval $[r_{i,j}, d_{i,j}]$ into a set of equal length time slots of size ϵ. Then we consider a new version of the problem in which if the job j is executed by the processor i, for every such time slot, either j is executed during the whole slot or is not executed at all during that slot. We are able to show that the value of the optimal solution for the new problem is at most an additive factor of ϵ far from the optimal solution of our original problem.

A configuration c is a schedule for a single job on a single processor. Specifically, a configuration determines the slots (as we described above) during which one job is executed. Let $\mathcal{C}_{ij}$ be the set of all possible feasible configurations for all jobs in all processors. We introduce the binary variable $x_{i,j,c}$ that is equal to one if the job $j \in \mathcal{J}$ is entirely executed on the processor $i \in \mathcal{P}$ according to the configuration c, and zero otherwise. Note that, given the configuration and the processor i where the job j is executed, we can compute the energy consumption $E_{i,j,c}$ for the execution of j. With the time discretization, we obtain a set of discrete time points based on which we define a set of continuous non-overlapping intervals $\mathcal{I}$ such that exactly one job is executed during each interval. We say $I \in (i, j, c)$ if the interval $I \in \mathcal{I}$ is included in the configuration c of processor $i \in \mathcal{P}$ for the job $j \in \mathcal{J}$. Then, we obtain the following integer program.

$$\min \sum_{i,j,c} E_{i,j,c} \cdot x_{i,j,c}$$

$$\sum_{i,c} x_{i,j,c} \geq 1 \qquad \forall j \in \mathcal{J} \qquad (1)$$

$$\sum_{(i,j,c):I\in(i,j,c)} x_{i,j,c} \leq 1 \qquad \forall I \in \mathcal{I} \qquad (2)$$

$$x_{i,j,c} \in \{0,1\} \qquad \forall i \in \mathcal{P}, j \in \mathcal{J}, c \in \mathcal{C}_{ij} \qquad (3)$$

Inequality (1) enforces that each job is entirely executed according to exactly one configuration. Inequality (2) ensures that at most one job is executed in each interval. We next relax the constraints (3) such that $x_{i,j,c} \geq 0$. We claim that the relaxed LP can be solved in polynomial time with the Ellipsoid algorithm because it admits polynomial time separation oracle but we omit this claim due to space constraints. So, we propose the following randomized rounding algorithm for obtaining a feasible integral solution.

Algorithm 1

```
1: Solve the configuration LP relaxation.
2: For each job $j \in \mathcal{J}$, choose a configuration at random with probability $x_{i,j,c}$.
3: For each interval $I$, scale the speeds of the jobs executed during $I$ so that they are executed with
   equal speeds during $I$.
```

Theorem 1 *There is an approximation algorithm which achieves a ratio of* $(1+\varepsilon)B_\alpha$ *for the heterogeneous multiprocessor speed-scaling problem without migrations in time polynomial to* n *and* $1/\varepsilon$.

References

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

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

[3] M. Andrews, A. F. Anta, L. Zhang, and W. Zhao. Routing for power minimization in the speed scaling model. *IEEE/ACM Trans. on Networking*, 20:285–294, 2012.

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

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

[6] A. R. Brodtkorb, C. Dyken, T. R. Hagen, J. M. Hjelmervik, and O. O. Storaasli. State-of-the-art in heterogeneous computing. *Sci. Program.*, 18:1–33, 2010.

[7] G. Greiner, T. Nonner, and A. Souza. The bell is ringing in speed-scaled multiprocessor scheduling. In *SPAA*, pages 11–18. ACM, 2009.

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

[9] A. Wierman, L. L. H. Andrew, and A. Tang. Power-aware speed scaling in processor sharing systems. In *INFOCOM*, pages 2007–2015, 2009.

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