Parallel Computing 38 (2012) 615–630

Contents lists available at SciVerse ScienceDirect

Parallel Computing

journal homepage: www.elsevier.com/locate/parco

Parallel job scheduling for power constrained HPC systems

M. Etinski *, J. Corbalan, J. Labarta, M. Valero

Computer Science Department, Barcelona Supercomputing Center, Barcelona, Spain
Department of Computer Architecture, Technical University of Catalonia, Barcelona, Spain

ARTICLE INFO

Article history:
Received 9 September 2011
Received in revised form 30 July 2012
Accepted 13 August 2012
Available online 4 September 2012

Keywords:
Power budget
CPU power consumption
Parallel job scheduling

ABSTRACT

Power has become the primary constraint in high performance computing. Traditionally, parallel job scheduling policies have been designed to improve certain job performance metrics when scheduling parallel workloads on a system with a given number of processors. The available number of processors is not anymore the only limitation in parallel job scheduling. The recent increase in processor power consumption has resulted in a new limitation: the available power. Given constraints naturally lead to an optimization problem. We proposed *MaxJobPerf*, a new parallel job scheduling policy based on integer linear programming. Dynamic Voltage Frequency Scaling (DVFS) is a widely used technique that running applications at reduced CPU frequency/voltage trades increased execution time for power reduction. The optimization problem determines which jobs should run and at which frequency. In this paper, we compare the *MaxJobPerf* policy against other power budgeting policies for different power budgets. It clearly outperforms the other power-budgeting approaches at the parallel job scheduling level. Furthermore, we give a detailed analysis of the policy parameters including a discussion on how to manage job reservations to avoid job starvation.

1. Introduction

Power consumption has become the primary concern in modern high performance computing (HPC) system design. The high power draw of modern systems leads to high operating costs, reliability issues and environment protection concerns. However, these are not the only factors that may limit available power in HPC environments. A power budget may be imposed by the existing power provisioning facilities as well as by any of the high power consumption issues mentioned before. It should be mentioned that power provisioning infrastructure is extremely costly. The cost of building a power provisioning facility ranges of $10–22 per deployed IT watt [2]. Nowadays, large scale computing centers typically consume few megawatts of power. According to some predictions, future exascale systems might consume more than 100 megawatts [14]. It is expected that in the future, the main goal of HPC, performance, will be limited even more by the available power.

CPU power presents a significant portion of total system power when the system is under load [12]. Therefore, in this work we aim to maximize job performance in an HPC center under a given CPU power budget. The budget is given for all system processors. We have shown before that the DVFS technique can improve job wait times in a case of a power constrained system [5]. DVFS (Dynamic Voltage Frequency Scaling) is a widely used technique that trades processor performance for lower power consumption. With DVFS a processor can run at one of the supported frequency/voltage pairs lower than the nominal one. Lower frequency and voltage lead to significantly lower power consumption allowing more jobs to run

* Corresponding author at: Computer Science Department, Barcelona Supercomputing Center, Barcelona, Spain.
E-mail addresses: maja.etinski@bsc.es (M. Etinski), julita.corbalan@bsc.es (J. Corbalan), jesus.labarta@bsc.es (J. Labarta), mateo.valero@bsc.es (M. Valero).

0167-8191/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.parco.2012.08.001

simultaneously. In spite of the job runtime increase due to frequency scaling, overall performance measured in job performance metrics improves with DVFS application because of shorter job wait times.

In traditional parallel job scheduling the available number of processors has been considered to be the only limiting factor preventing all queued jobs to start immediately. With the increase in processor power consumption, the available power has emerged as a new constraint. A policy that assigns CPU frequency to a job selected for execution in such a way that a given power budget is not violated has already been proposed [5]. Here we refer to this policy as *PB-guided*. It presents a power budgeting upgrade of the traditional EASY backfilling policy [28]. When the EASY scheduler is invoked, it selects jobs to run on the available processors from the wait queue one by one. Similarly, the *PB-guided* policy assigns CPU frequency to each job independently of the frequencies to be assigned to the other jobs that will be selected for execution. The *PB-guided* policy is designed to manage power in a conservative way in order to save some power for the other jobs selected for execution. Due to this conservatism, the power budget may not be fully exploited.

In this paper we analyze in more detail a more sophisticated power budgeting parallel job scheduling policy, *MaxJobPerf*, based on integer linear programming [6]. Each time the scheduler is invoked, it solves an optimization problem to determine which jobs from the wait queue should start. The same optimization problem distributes the available power among the jobs assigning CPU frequency to each of the selected jobs. In this way the scheduler allocates both types of available resources, the processors and power, to all jobs at the same time. Linear programming has been already investigated for power unconstrained systems targeting specific types of scheduling such as scheduling of moldable jobs [4] or scheduling in heterogeneous environments [29].

We have evaluated the *MaxJobPerf* policy against an EASY backfilling based baseline without DVFS and the *PB-guided* policy. The policies have been tested for five workloads from production use with up to 9216 processors. The results clearly show that the *MaxJobPerf* policy outperforms the other two policies for all observed workloads. Due to its maximal use of the power budget and a relaxed constraint of job execution order, the *MaxJobPerf* significantly reduces the average job wait time, even when compared to the *PB-guided* policy. The average job wait time is even few times better for some workloads. The reduction in the average wait time leads to further improvements in other job performance metrics. The evaluation has been done for different power budgets and the *MaxJobPerf* policy performs consistently better than the other approaches over various power budgets.

As we showed before [6], for some workloads it is of crucial importance that the *MaxJobPerf* scheduler manages job reservations. This is mandatory in order to control the job wait time. Without reservations, large jobs requesting a significant portion of the machine can experience very high wait times. In this paper, we performed an extensive study to determine an appropriate reservation assignment condition. The performed simulations show that wait time based reservations give better results than job size based reservations. We investigated how to determine an appropriate wait time threshold depending on the system load.

The rest of the paper is organized as follows. The next section gives a description of the *MaxJobPerf* policy. In Section 3 we explain the models of frequency scaling impact on the job runtime and the average power consumption. The first part of Section 4 presents the evaluation methodology whilst the rest of the section discusses the obtained results. Section 5 gives an overview of the related work. It is followed by our conclusions.

2. Parallel job scheduling for a given power budget

A parallel job scheduling policy determines the order in which jobs submitted to a supercomputing center will be executed. In this work we investigate on-line scheduling for asynchronous job arrivals. When there are enough resources for all arriving jobs to run simultaneously, the problem is trivial. A more complicated case appears when there are more requested than available resources. The simplest policy, First Come First Served (FCFS), schedules jobs according to their arrival order. Once there are enough processors for the first job in the wait queue, the first job starts with its execution. Various backfilling policies have been proposed to improve system utilization while keeping the FCFS order with some exceptions. For instance, with the widespread EASY backfilling policy, a job can violate FCFS order if it does not delay the reservation made for the first job in the wait queue. In this way, smaller jobs can be scheduled before a previously arrived bigger job. With the backfilling policies a user is obliged to submit the requested time of the job being submitted in addition to the requested number of processors. The requested time presents a user estimate of the job runtime. It is in the user's interest to provide accurate estimates as a job with underestimated requested time will be killed and an overestimation may lead to longer wait time. Having job requested times, the scheduler can estimate job finish times and deal with job reservations needed for the backfilling policies. With power becoming an important resource, parallel job scheduling gets an additional dimension. While selecting jobs from the wait queue for execution, a power-aware scheduler should be aware of available power over the requested time in addition to available processors.

2.1. The MaxJobPerf policy

Having these constraints, such as available power and processors, leads naturally to an optimization problem. We propose a power budgeting parallel job scheduling policy based on integer linear programming. With the *MaxJobPerf* policy a job can be scheduled for execution in two ways. If there are enough resources to run a job at its arrival time, the job will start imme-