J. Parallel Distrib. Comput. 72 (2012) 579–590

Contents lists available at SciVerse ScienceDirect

J. Parallel Distrib. Comput.

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

Understanding the future of energy-performance trade-off via DVFS in HPC environments

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 19 July 2011
Received in revised form
17 January 2012
Accepted 20 January 2012
Available online 28 January 2012

Keywords:
DVFS
Energy efficiency
High performance computing

ABSTRACT

DVFS is a ubiquitous technique for CPU power management in modern computing systems. Reducing processor frequency/voltage leads to a decrease of CPU power consumption and an increase in the execution time. In this paper, we analyze which application/platform characteristics are necessary for a successful energy-performance trade-off of large scale parallel applications. We present a model that gives an upper bound on performance loss due to frequency scaling using the application parallel efficiency. The model was validated with performance measurements of large scale parallel applications. Then we track how application sensitivity to frequency scaling evolved over the last decade for different cluster generations. Finally, we study how cluster power consumption characteristics together with application sensitivity to frequency scaling determine the energy effectiveness of the DVFS technique.

1. Introduction

Energy efficiency has become one of the most critical issues in modern cluster design because of very high operating costs, reliability issues and environmental concerns. Since CPU power accounts for a large portion of total system power consumption [11], there has been considerable research on CPU power management. The majority of these works are based on the DVFS technique (Dynamic Voltage Frequency Scaling). Processors that support DVFS can run at a lower frequency/voltage setting consuming less power. Unfortunately, lower frequency settings generally lead to longer execution times.

DVFS energy saving techniques in HPC (High Performance Computing) systems can be classified into two classes. The first class of approaches accepts a certain penalty in performance for reduced energy consumption [29,14,16,21]. The other class runs processors at lower frequency only if they are not on the critical path avoiding performance loss [18,22,28,20]. There are two main drawbacks of the second class of approaches: they can be applied only to specific applications (i.e. load imbalanced or communication intensive) and they involve fine grain DVFS use that may present a chip reliability issue. In this paper we target the first class of approaches discussing DVFS potentials for the energy-performance trade-off in current and future large scale HPC clusters.

Fig. 1 gives two power/execution time scenarios. The first one represents an application execution at the nominal CPU frequency whilst the other case assumes that the application runs at a reduced frequency f. In the second case the application takes longer, finishing at the moment T_2. When running at the nominal frequency the application execution ends at the moment T_1. In this case the system consumes power $P(f_{\max})$ until the moment T_1 and P_{idle} from T_1 until T_2. The application running at the reduced frequency dissipates $P(f)$ over the entire observed time interval. The mentioned values are the *average system power* consumption values over the observed intervals. Hence, the energy $E(f_{\max})$ consumed in the first case is $P(f_{\max}) * T_1 + P_{\text{idle}} * (T_2 - T_1)$. In the second case, the energy consumption $E(f)$ is equal to $P(f) * T_2$. Since CPU power consumption accounts for a high portion of the total system power (50% of system power under load [11]), reduction in CPU power due to frequency scaling leads to significant difference between $P(f_{\max})$ and $P(f)$ ($P(f) < P(f_{\max})$). Therefore, the second scenario in which the application runs at reduced frequency has been considered to be more energy efficient ($E(f) < E(f_{\max})$).

New attitudes, contrary to conventional wisdom that in general DVFS saves energy in spite of performance loss, have emerged recently. For instance, Le Sueur et al. found that while DVFS was effective on older platforms, it actually increases energy usage of *sequential* applications on the most recent platforms [19].

Running an application at lower frequency/voltage results in significantly lower CPU power consumption. However, due to an increase in the application execution time, frequency reduction may lead to higher energy consumption. Critical aspects that must

* 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).

0743-7315/$ – see front matter © 2012 Elsevier Inc. All rights reserved.
doi:10.1016/j.jpdc.2012.01.006

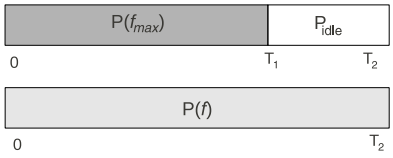

Fig. 1. Two energy scenarios.

be considered when evaluating frequency scaling potentials for energy saving are the following:

- The increase in the execution time for a given amount the frequency was reduced by.
- The portion of total system power reduction for a given amount the frequency was reduced by.
- The ratio of idle and active system power.

Longer execution time at lower frequency is not only a performance issue but it determines whether the reduction in frequency results in energy savings. The increase in the execution time is not necessarily proportional to the reduction in frequency. How much frequency scaling affects the application execution time depends on non-CPU activity i.e. memory accesses and communication latency. It is important to state that this work targets large scale parallel applications whose performance loss highly depends on the portion of time spent in communication as shown in Section 2.

Obviously, the amount by which total system power can be reduced is one of the parameters that determine energy efficiency of the DVFS technique. CPU power reduction is limited by the constantly increasing portion of leakage power and the voltage-scaling window. Furthermore, the amount by which total system power can be reduced depends on the CPU power fraction in total system power.

When evaluating an energy saving approach it is common to regard only the energy consumed during an application execution even when different approaches do not have the same execution times. Miyosi argued that the power consumed while idle must be taken into account if overall saving is the goal [24]. The system cannot be simply turned off when an application finishes. In fact idle cluster power is still very high accounting for about half of the power consumed under load [8]. Idle processors can be put into a low power mode but this is still not the case with other system components. Future cluster design must radically decrease idle power in order to achieve energy proportional computing [1]. Thus, two energy scenarios must be compared during the *same* time interval.

Our contributions in this paper are:

- We proposed and evaluated a model of frequency scaling impact on execution time for *large scale MPI applications*.
- Frequency scaling impact on performance was measured and analyzed on a modern platform for real world applications with up to 712 processors.
- Application sensitivity to frequency scaling was compared over different cluster generations.
- A parametric analysis of DVFS energy efficiency was performed for large scale parallel applications.

Similarly to findings of Le Sueur et al. for sequential applications, we find that the DVFS technique potentials for parallel applications are diminishing as well, in spite of the fact that the communication time does not scale with frequency scaling. Execution times of parallel applications running on newer systems tend to be more sensitive to frequency scaling than they were before. Though energy-proportional computing is still a research challenge, we show how the eventual reduction in idle power consumption will further diminish opportunities for DVFS energy savings.

In spite of decreasing DVFS energy saving potentials, the technique still can be used to reduce power consumption in power constrained systems to run more jobs simultaneously [7], resulting in the same or higher energy consumption. Because of increasing main memory power consumption, memory DVFS has been proposed recently [5,4]. Applying frequency/voltage scaling to both processors and memory subsystem might present a solution for future clusters.

The rest of the paper is organized as follows. The next section explains how frequency scaling affects the application execution time. Section 3 gives a parametric analysis of application and platform parameters that determine whether a reduction in CPU frequency leads to a more energy efficient execution. Section 4 presents related work. Our conclusions are given in Section 5.

2. CPU frequency scaling impact on execution time

2.1. The β metric

Running an application at reduced frequency increases the application execution time. The increase in execution time is not necessarily proportional to the reduction in frequency. An application sensitivity to frequency scaling is determined by its CPU-boundness. Hsu and Kremer have introduced the β metric that gives application slowdown compared to the CPU slowdown [15]. They have used it for sequential applications accounting for memory access time that stays the same with frequency scaling. The ratio between $T(f)$, the application execution time at frequency f and $T(f_{\max})$, the execution time at the nominal frequency equals:

$$\frac{T(f)}{T(f_{\max})} = \beta\left(\frac{f_{\max}}{f} - 1\right) + 1. \tag{1}$$

This relation between the reduction in frequency and the increase in execution time is derived assuming that the computation time T_{CPU} scales inversely proportional with frequency while the memory access time T_{MEM} does not change:

$$\frac{T_{\text{CPU}}(f) + T_{\text{MEM}}}{T_{\text{CPU}}(f_{\max}) + T_{\text{MEM}}} = \frac{T_{\text{CPU}}(f_{\max}) * \frac{f_{\max}}{f} + T_{\text{MEM}}}{T_{\text{CPU}}(f_{\max}) + T_{\text{MEM}}} \tag{2}$$

$$\frac{T(f)}{T(f_{\max})} = \frac{T_{\text{CPU}}(f_{\max})}{T_{\text{CPU}}(f_{\max}) + T_{\text{MEM}}}\left(\frac{f_{\max}}{f} - 1\right) + 1. \tag{3}$$

Thus, β equals:

$$\beta = \frac{T_{\text{CPU}}(f_{\max})}{T_{\text{CPU}}(f_{\max}) + T_{\text{MEM}}}. \tag{4}$$

Different applications experience different execution time penalties depending on their CPU-boundedness. Theoretically, if an application would be completely CPU bound, its β would be equal to 1 while $\beta = 0$ means that the execution time is insensitive to frequency scaling. In practice, the β parameter has values between the extremes 0 and 1. It is important to mention that the β parameter describes application/platform characteristics. According to Eq. (4) β of an application should not depend on the amount the frequency was reduced by. A previous work has assessed the variance between two beta values of an applications computed with different reduced frequencies. This work showed that the highest variance for various applications was 5% on the platforms used [9].

In the case of parallel applications, communication latency is an additional non-CPU activity insensitive to frequency scaling. The same metric β is used for parallel applications to measure their performance sensitivity to frequency scaling [9,7]. In the next section we report large scale parallel application β values computed using our measurement results.