Energy Optimization of Parallel Workloads on Unreliable Hardware

Chhaya Trehan, Hans Vandierendonck, Georgios Karakonstantis, Dimitrios S. Nikolopoulos
Queen's University of Belfast
Email: {c.trehan, h.vandierendonck, g.karakonstantis, d.nikolopoulos}@qub.ac.uk

***Abstract*—We present a *work in progress* report on the problem of minimizing the energy consumption of a parallel application on an unreliable hardware platform, specifically unreliable memory. In such a system, not all of the application data in memory is accurate at all the times. Allowing some inaccuracies in the data saves memory refresh power at the cost of increased inaccuracy which in many application domains can be dealt with algorithmic error resilience. We are building an analytical model to capture the CPU energy consumption of a parallel application with precedence constraints running on a system with unreliable memory. Using this model, we plan to provide a framework for analytically selecting CPU frequencies that minimize the overall CPU energy consumption of the application.**

***Keywords*-Energy Optimization; Global DVFS; Approximate Computing;**

I. Introduction

Most memory systems in the contemporary computing platforms choose to refresh at a rate commensurate to the worst case retention time of a memory cell to guarantee hundred percent accuracy [1]. Many applications that run on these systems on the other hand can deal with some inaccuracies in the stored data without much degradation in their quality metrics. There is thus scope for reducing the power of a memory system by reducing its refresh rate. This trade off between power and reliability of a memory system opens up space for research on the design of memory systems that can hit the right spot by maximizing the energy savings while satisfying the workload demands such as expected accuracy of the final output. A potential point in this two dimensional design space is that of a hybrid memory system that is divided into regions with varying degrees of reliability. The more reliable regions provide better guarantees in terms of the accuracy of the stored data at the cost of extra access delays. An application scheduled to run on a system with a hybrid memory system expereinces varying amounts of access latencies depnding upon what region of memory is accessed. These access latency variations in turn affect the energy consumption of the CPU as it waits for the desired data. The CPU energy optimization techniques that save energy by decreasing the operating frequencies of the cores at the cost of an increased delay need to be tuned to account for the memory access latencies. Precisely accounting for the memory access delays of the application helps exploit the slack between the time to completion and the given performance budget.

The existing analytical models for the global DVFS [2] do not account for the time overhead of the access latency of memory, which leads to an imprecise estimate of the slack and hence can lead to an over optimistic selection of operating frequencies. The same holds true for a hybrid memory system. This calls for an analytical model that not only accounts for the performance and hence the energy overhead of memory accesses made during the execution an application, but also distinguishes between the overheads incurred by the accesses to the reliable and unreliable parts of the memory system. We present one such model. Given a schedule for an application, our model estimates its energy consumption and performance in terms of the characteristics of the application and its schedule such as the number of CPU cycles required to run each task, the number of memory accesses made by each task and the amount of parallelism. The ultimate goal of our work is to build an analytical framework to predict the operating frequencies for global DVFS to minimize the overall CPU energy consumption of a given application.

II. Related Work

A common approach to reduce the energy consumption of an application is to reduce the operating frequency of the cores at the cost of increased executon time. Most of the theoretical work on the energy-delay trade off deals with the *local* dynamic voltage and frequency scaling, where every core's voltage and frequency can be set separately [2]. We study the problem of energy minimization under a performance constraint using global DVFS where the voltage and frequency are set for the entire chip.

In their recent paper, Gerards et al [2] show that using a single clock frequency during the execution of a *parallel* application with precedence constraints does not lead to optimal energy consumption and present an approach for varying the frequency during execution to minimize energy. Li in his pioneering work [3] presents heuristic algorithms for energy optimization that treat scheduling and frequency selection as two independent subtasks performed one after the other. Further, Gerards et al [2] show that the tasks of determining a schedule and frequencies that together minimize the energy consumption should not be considered separately and study the relation between the two. They define a scheduling criterion for energy optimization and show how to determine frequencies that minimize energy

consumption. They characterize a schedule in terms of *parallelism*, which gives for each number of cores the number of clock cycles for which exactly that many cores are active.

III. System and Application Model

A. Application

We consider an application running on a multicore processor. The application itself consists of a set T of N tasks, denoted by $T_1, ...T_N$. We define a binary relation $\prec$ on T to represent the precedence constraints. For any two tasks T_i and T_j in T, $T_i \prec T_j$ means that T_i has to be completed before T_j starts. We consider an overall deadline t_{budget} for the entire application. A task T_i is characterized by two attributes, namely: the *compute workload:* cw_i and the *data workload:* dw_i. The compute work load is the number of clock cycles required to perform the computations of the task. The data workload is the number of memory accesses a task has to make during its execution. We assume an application wide parameter called *data to compute quotient* d which is the ratio of data to compute workloads of the application. It can be viewed as the number of memory accesses per CPU instruction of the application For a task T_i with compute workload of cw_i, its data workload can be inferred as the product of cw_i and d. The application can be depicted as a labeled DAG (Directed Acyclic Graph) where nodes represent the tasks and the (*Directed*) edges represent the precedence constraints. Each node carries a label depicting the CPU workload cw_i of the associated task. To account for the fact that not all the memory accesses of a task bear the same delay in a hybrid memory system, we define another application wide parameter called the *reliability quotient* r which is the relative number of memory accesses made to the reliable portion of the memory.

B. Computing Platform

The Application runs on a Chip Multiprocessor system with $M > 1$ homogeneous processing cores. All the cores have similar capabilities and run at the same frequency. we envisage a memory where a portion of memory is almost hundred percent accurate and stores data *exact data*. The rest of the memory is configured to store *data that may be approximated*.

IV. Energy Consumption and Performance Model

We extend the energy and performance model presented in [2]. The two main ideas that we borrow from [2] are that using a constant clock frequency for an interval during which a fixed number of cores are active is optimal in terms of energy consumption and that the overall energy consumption of an application can be expressed in terms of the amount of parallelism In interest of brevity, we refer the reader to go through [2] to fully appreciate the concept of power modeling in terms of parallelism. For an application with N tasks running on a processor with M cores, its amount of parallelism for a given schedule is defined as a vector $[w_1, w_2, ...w_m...w_M]$, where w_m is the total number of CPU cycles for which exactly m cores are active. Using the idea that a constant frequency for a fixed number of cores (parallelism) leads to an optimal energy consumption, the task of global DVFS for energy optimization is reduced to finding a vector $f = [f_1, f_2, ..f_m...f_M]$ of frequencies for the following optimization problem:

$$\begin{aligned} \underset{f_1, f_2, ... f_m}{\text{minimize}} \quad & \sum_{m=1}^{M} [\overline{p}_m(f_m) w_m] \\ \text{subject to} \quad & \sum_{m=1}^{M} [\frac{w_m}{f_m}] \leq t_{budget} \end{aligned}$$

where $\overline{p}_m$ is energy consumed per cycle at a frequency f_m.

For a given amount of parallelism w_m, $w_m d$ accesses to memory are made. The CPU keeps clocking at a frequency f_m for the duration of these $w_m d$ memory accesses. If t_a is the latency of memory accesses, $(w_m d)t_a$ is the duration for which the CPU waits for memory accesses. The additional cycles expended per core on memory accesses for w_m is thus $w_m d t_a f_m$. Replacing w_m with $w_m + w_m d t_a f_m$ in the above optimization problem leads to a new optimization problem that accounts for the CPU energy consumed not only on the actual CPU work done but also the addition clock cycles expended on waiting for the memory accesses. One can easily refine this optimization problem for a system with hybrid memory by incorporating the *reliability quotient* and different access delays for reliable and unreliable memory accesses.

V. Conclusion and Road Map

We have presented a new model for energy optimization on a multicore system. As a next step, we plan to use this model to come up with analytical formulae for determining the operating frequencies $f_1, f_2, ...f_M$ for energy minimization. We plan to use these formulae on a variety of real benchmarks and estimate the potential energy savings.

References

[1] S. Liu, K. Pattabiraman, T. Moscibroda, and B. G. Zorn, "Flikker: Saving dram refresh-power through critical data partitioning," *SIGPLAN Not.*, vol. 47, no. 4, pp. 213–224, Mar. 2011. [Online]. Available: http://doi.acm.org/10.1145/2248487.1950391

[2] J. L. H. Marco E.T. Gerards and J. Kuper, "On the interplay between global dvfs and scheduling tasks with precedence constraints," *IEEE TRANSACTIONS ON COMPUTERS*, vol. 64, no. 06, 2015.

[3] K. Li, "Scheduling precedence constrained tasks with reduced processor energy on multiprocessor computers," *Computers, IEEE Transactions on*, vol. 61, no. 12, pp. 1668–1681, Dec 2012.