Global optimization model on power efficiency of GPU and multicore processing element for SIMD computing with CUDA

Da-Qi Ren · Reiji Suda

Abstract Estimating and analyzing the power consuming features of a program on a hardware platform is important for energy aware High Performance Computing (HPC) optimization. It can help to handle design constraints at the level of software, choose preferable algorithms in order to reach the best energy performance. Optimizing the power efficiency of CUDA program on GPU and multicore processing element is a problem in combinatorial optimization because of the complexity of power factors and criteria. A four-tuple global optimization model has been created to indicate the procedure to find optimal energy solution. In addition, an experimental method is illustrated to examine SIMD computing for capturing power parameters, five individual energy optimization methods are provided and implemented. The optimization results have been validated by comparative analysis on real systems.

Keywords Energy aware HPC · GPGPU computing

1 Introduction

Graphics Processing Units (GPU) support many advanced HPC solutions because of the efficiency in massively parallel processing. GPU and multicore architecture becomes a major choice in constructing modern supercomputers and accelerating various of scientific computing applications. However, the power consumption of such architecture has been continually increasing, and much less research has been carried out to optimize the power performance for GPU and multicore Processing Element (PE) with integrated parallel programming paradigms.

Related works There are many research efforts on CPU power models at different levels of abstractions. A power model for CUDA program on GPU and multicore structure can be created following some of the CPU design principles. In lower level models the available physical information allows acquiring accurate power estimates, such as the measure of the amount of energy expended by a microprocessor for each instruction that the microprocessor executes [1]. An integrated power model is proposed in [2] for a GPU-based computer to predict execution times and calculate dynamic power events, by integrating an analytical timing model and an empirical power model, power consumption of GPGPU workloads is predicted with only the instructions. A statistical approach using the GPU performance counters for estimating power consumption of GPU kernels is presented in [3], linear correlation between the performance profiles and power consumption by statistical model learning is investigated. Higher level models use more indirect and approximate design parameters, such as the algorithm level power model that we have introduced in [4, 5]. We directly measure Single Instruction Multiple Data (SIMD) instruction flow over each component because SIMD computing has the character that each component executes its instruction sequence repeatedly. The energy consumption of a CUDA program depends not only on each single instruction but also on the surrounding code and the environment. The advantage of the approach in [4, 5] is that instruction mixture...
information, pipelining structure and out of order processing
information can be covered in the SIMD flow that are
measured. Therefore a fixed sequence of assembly code will
take approximately a constant number of CPU/GPU clock
cycles, the energy consumptions will be evaluated accord-
ingly.

Energy aware algorithms have been designed and im-
plemented based on the power model that contribute en-
ergy savings from different fronts including hardware se-
lection, component utilization, and numerical methods. To
reach the overall optimum power efficiency, global opti-
mization concept has been imported to this work [6]. We
have created a global energy optimization model that spec-
ifies all major performance constrains, dependencies and
their interactivities. We also present a flow chart to indi-
icate the scenario for finding the best alternative among
all feasible energy solutions. Finally the global energy
optimization methodology is tested and validated by ex-
amining the C/CUDA programs executing on real sys-
tems.

2 Architecture and power model of CUDA PE

A CUDA Processing Element is defined as a hardware
unit including three major components, i.e. GPU, multicore,
memories and PCI bus on the same main board, that exe-
cutes streams of CUDA kernel instructions [5, 7]. CUDA
PE allows efficient program implementations on each of
the components for computing intensive tasks. Single In-
struction Multiple Data (SIMD) is a parallel programming
model that runs tasks simultaneously in order to execute re-
results faster. Multicore and GPU provide cooperative archi-
tectures on which both SIMD and single program multiple
data (SPMD) programming models can co-exist and com-
plement each other. Energy consumption is originated from
the hardware components. When a SIMD program is exe-
cuted, each component will perform a sequence of operation
repeatedly. If the component’s frequency is fixed and tem-
perature retains the same value, its power will only change
with the number of operations it executes in one time unit.

An experimental method to estimate the power consump-
tion of a CUDA PE is introduced in [5]. In the method,
power of each CPU, GPU and main board component in-
volved in the computation is measured firstly. The power
feature of the PE, in Flops/Watt, is obtained from the mea-
surements, which shows the rate of computation that can
be provided by the PE for every watt of power consumed.
Therefore to a specific problem, the total execution time can
be estimated from the total workload and the computation
speed, the energy consumption will be evaluated from the
dynamic power and execution time.

2.1 Power model

As the power dissipation of a CUDA program is originated
from each power consuming component, the PE’s power can
be modeled as

\[ P_{\text{system}}(w) = \sum_{i=1}^{n} P_{G_PU}^{i}(w_i) + \sum_{j=1}^{m} P_{CPU}^{j}(w_j) + P_{\text{mainboard}}(w) \]

(1)

Where \( P_{\text{system}} \), \( P_{CPU} \), \( P_{GPU} \) and \( P_{\text{mainboard}} \) represent
the power of the overall system, GPU, CPU and mainboard, re-
spectively. \( N \) and \( M \) are the numbers of GPUs and CPUs
that involved in the computing of workload \( w \). \( w_i \) and \( w_j \)
represent the workload assigned to \( GPU_i \) and \( CPU_j \), re-
spectively. In Table 1 we list the software and hardware fac-
tors that impact the power consumption of each component
including the relative power parameters.

2.2 Power measurement

The instruments used for the measurement includes National
USB-6216 BNC data acquisition, Fluke i30s/i310s current
probes, and Yokogawa 700925 voltage probes. The room
was air-conditioned in 23°C. We use LabView 8.5 as oscil-
skopes and analyzer for result data analysis. Measurement
for the power consumption of a CUDA PE has been intro-
duced in [5]. Figure 1 shows the electrical circuit diagram of
the connections from a power supply unit (PSU) to each
power consuming component. It includes the theoretical in-
put values on each of the measurement points, a sample re-
sult and a picture of the real testing environment.

2.3 Hardware environment

In this work we use 2 CUDA PEs composed of different type
of GPU and multicore components: (1) One Intel QX9650
CPU and 2 GeForce 8800 GTS GPUs. The GPU has 12 MPs
(Multi-Processors) as computing cores, and each MP has 8
SPs (Streaming Processors) to execute threads. (2) Intel i7
CPU and Tesla C2050 GPU. The GPU has 448 CUDA Cores
that can deliver up to 515 Gigaflops of double-precision
peak performance, up to 6 GB of GDDR5 memory per GPU.

2.4 Example problem

Given \( A \in R^{n \times n}, B \in R^{n \times n} \), there are various algorithms
to implement \( C = AB \) with CUDA on GPU and multi-
core. In this paper we partition the matrix A to let

\( A = (A_1^T, A_2^T, \ldots, A_k^T) \). \( C = AB \) is calculated by parallel and