Statistical Modeling of Power/Energy of Scientific Kernels on a Multi-GPU system

Sayan Ghosh  
Dept. of Computer Science  
University of Houston  
Houston, Texas  
sgo@cs.edu

Sunita Chandrasekaran  
Dept. of Computer Science  
University of Houston  
Houston, Texas  
sunita@cs.uh.edu

Barbara Chapman  
Dept. of Computer Science  
University of Houston  
Houston, Texas  
chapman@cs.uh.edu

Abstract—Energy efficiency of GPUs has facilitated the usage of GPUs in many complex scientific applications. Nodes with multi-GPUs along with multi-core CPUs are quite common in today’s HPC landscape. This gives the flexibility to utilize CPUs or accelerators or even both according to the workload characteristics. It is not possible to measure power and energy accurately in all the cases, an alternate approach is to estimate power and energy using statistical methods. Apart from saving time and money, reasonable prediction of power/energy would lead to power saving optimizations for certain applications, without compromising performance. In this paper we employ parametric and non-parametric regression analysis to model power and energy consumption of some of the common high performance kernels (DGEMM, FFT, PRNG and FD stencils) on a multi-GPU platform. Our experiments show that using a minimal set of hardware counters and performance attributes, the average error between the measured and the predicted values of power and energy is only ∼ 4%.

Index Terms—Power, Energy, Energy Efficiency, Multi-GPU, Statistical Modeling

I. INTRODUCTION

To mitigate energy consumption, presently, off-chip hardware accelerators like GPUs are explored since they are capable of achieving substantial performance for arithmetic bound problems. Increasing the computing potential of a node (consisting of 2-8 GPUs) has also increased the power consumption significantly (1000 Watts and above). GPUs like Nvidia Fermi series consume more than twice the power as compared to multi-core CPUs. Hence, there is an onus on software optimization to bridge the gap in power by improving energy efficiency. We believe that the power/energy modeling of application kernels on heterogeneous (multi-socketed CPU and multi-GPU system on a single node) systems could lead to a better understanding of how the compiler/runtime can be exploited for optimizing the overall energy consumption factor. Both coarse (entire system, per node, rack) and fine grained (individual component level, e.g, CPU core, memory, bus) power measurements are indispensable if we need to gain deeper understanding of the total energy consumption of a system. We observed that regardless of different types of communication and computation patterns found in each of these kernels (as shown in Fig. 1), some common parameters such as global memory accesses and instruction throughput determine the overall power/energy consumption of a node, as shown in Fig. 1b. This observation helps us identify a limited set of hardware counters, that could explain the performance of an application kernel on a GPU, and ultimately, could predict power and energy consumption for a particular class of application. We have deliberately considered kernels belonging to various applications class since this introduces variations in our data set leading to a better prediction analysis. In our previous work [2], we explored a non-parametric regression approach, and observed reasonable prediction accuracy. This work motivated us to explore different statistical modeling techniques to understand the patterns in the data.

II. BACKGROUND

Data transfer is a major contributor to system energy for off-chip accelerators. We observe that some kernels achieve favorable performance per watt on multiple GPUs only when kernel computation time was considered (Fig. 2a). When data transfer time was included we noticed reasonable degradation in performance per watt, as shown in Fig. 2b.

A. Evaluation Platform

The testbed node consists of an AMD Magny-Cours Processor (6174) having a total of 24 physical cores. The node has 4 Nvidia Tesla M2050 GPUs, each with 448 compute cores. Fig. 3 shows the layout of the node, with the Yokogawa WT500 power analyzer. The node under test is found to have an idle power of 360 watts. A thread-safe API called

![Fig. 1. Performance of test kernels on 4 GPUs](attachment:fig1.png)

(a) Maximum Power consumption of test kernels on a 4 GPUs. The number on each of the bars indicate instructions per byte of memory access.

(b) Memory and Kernel execution times of the test kernels on 4 GPUs.
UHPwrLib was developed to remotely interface with the power analyzer, from either user program (by a function like Unix `gettimeofday()` called `get_power()`) or via a daemon. For a thorough analysis, it is required to sample power at frequent operating intervals of an application kernel.

**B. Software Tools and Test Kernels**

In our experiments we include both memory transfer time and kernel execution time while evaluating performance. Since, instantaneous power reading might be imprecise to judge the average power consumption of an application kernel, all the test cases are executed for many iterations to ensure that it reaches steady state power level. We use Nvidia CUDA FFT and BLAS libraries to perform Fourier transform and DGEMM on the GPUs, CUDA Visual Profiler to collect performance profiles of the GPU kernels, R statistical programming language [8] to apply regression analysis on GPU performance profiles. We discuss some of significant efforts made to optimize the test kernels.

- Isotropic FD stencil: We have used CUDA peer-to-peer transfer (for exchanging halo regions) to avoid a roundtrip to the host. We modify the kernel from CUDA SDK [7] for multi-GPUs.
- FFT: A single FFT could be expressed as a combination of many small FFTs i.e. we use CUFFT for the GPU implementation and this performs FFTs in concurrent batches (rather than one large FFT execution).
- DGEMM: CUDA streams were used in GEMM to overlap computation and communication (since Fermi class GPUs could overlap kernel execution and a bi-directional transfer between host and GPU) that gave 4% improvement (in total execution time) over the base version.
- Mersenne Twister relies on a set of parameters, that apart from the initial seed needs to be different for each thread to create a set of diverging random numbers. Input parameters could be shared by all the threads thereby giving us the opportunity to use faster GPU shared memory, thereby completely avoiding DRAM accesses and leaving only bit-wise arithmetic computations. We modify the kernel from CUDA SDK [2] for multi-GPUs.

**III. Statistical Modeling Techniques**

In this section, we will discuss some of the statistical terminologies that we will be using throughout the rest of the paper.

**A. Regression**

Linear Regression is the first type of regression analysis to be extensively used in scientific applications owing to its simplicity. This is because variables in a model that depend linearly on their unknown parameters are easier to determine than models which have non-linear relationships. It is represented by modeling the relationship between a scalar dependent variable (also called response variable) $Y$ and one or more explanatory variables (or independent variables or predictors, these terms are used interchangeably) denoted by $X$.

$$Y = \beta \ast X + \epsilon$$

where $\epsilon$ is the error amount, and $\beta$ represents the regression coefficients. Predictors like linear or polynomial regression are global models, where a single predictive formula is expected to hold the entire data space. When the data has features interacting in non-linear ways, assembling a single global model can be quite challenging. In such cases, **additive models** could be considered as an improvement over linear regression, where non-parametric functions are used to fit models locally and combine the results.

**B. Variable Selection**

It is often not required to consider all the available independent variables (predictors) in the model, since not every one of them contribute in predicting a dependent variable. Even worse, unnecessary predictors could add noise to the estimation, in addition to the computation cost. Hence, choosing a subset of the variables that has an impact on power/energy is an important preprocessing step. When the number of variables is not large, it is feasible to evaluate all the models. But for our case, since the number of independent variables is 8 for both power and energy prediction, total number of evaluations to be performed is $2^8 = 256$, which calls for considering an automatic scheme to choose predictor variables relevant to the

![Fig. 3. Testbed node with WT500 Power Analyzer](image-url)
model. We consider the Akaike Information Criterion (AIC), which is a measure of the relative goodness of fit of a statistical model. It favours smaller residual error in the model, but penalizes for including further predictors and helps avoiding overfitting\(^1\). The model with the lowest AIC is usually the best. The following variables were selected for our linear regression model, based on their AIC criteria.

\[ P \text{ or } E \sim TT + II + L2M + DR + DW + TH \]

The \( \sim \) in statistical parlance should be read "is modeled by". The acronyms representing independent/dependent variables are listed in Table I and would be used throughout this paper.

**TABLE I**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>TT</td>
<td>Execution Time</td>
</tr>
<tr>
<td>II</td>
<td>Instructions Issued (without replays)</td>
</tr>
<tr>
<td>TH</td>
<td>Operations per second</td>
</tr>
<tr>
<td>WSM</td>
<td>Warps executed per SM</td>
</tr>
<tr>
<td>ISM</td>
<td>Threads executed per SM</td>
</tr>
<tr>
<td>DR</td>
<td>DRAM reads</td>
</tr>
<tr>
<td>DW</td>
<td>DRAM writes</td>
</tr>
<tr>
<td>L2M</td>
<td>L2 Cache misses</td>
</tr>
<tr>
<td>P</td>
<td>Power consumed</td>
</tr>
<tr>
<td>E</td>
<td>Energy consumed</td>
</tr>
</tbody>
</table>

**IV. STATISTICAL MODELING AND ANALYSIS**

We perform linear and non-linear statistical modeling to predict power/energy consumption on the multi-GPU node, using performance attributes of kernels like DGEMM, FFT, 3D Finite Differences and Mersenne Twister PRNG. We have the same number of observations (performance on 1-4 GPUs) for each set (kernels). In our case, the dependent variable being modeled is power/energy, and the independent variables are the counters (collected by Nvidia CUDA Profiler) and performance attributes (such as time taken, GFLOPs) that have a strong correlation with power/energy.

**A. Linear Regression**

We use classical linear regression and select predictor variables using the AIC metric (Section III). Only resampling (splitting data set into 5 parts and performing analysis on one subset and validating on another, i.e. 5-fold cross-validation) is insufficient to judge a model. Hence we consider an external testing set for predicting out-of-sample data that were not part of the training set (75% of the dataset). Fig. 4 shows the predicted vs measured values of both energy and power using 5-fold cross-validation and using an external testing set. We observe that removing "DW" (DRAM Writes) improves the prediction accuracies for both power and energy estimation. At this stage, the power and energy models are represented as:

\[ P \sim II + DR + L2M + TH \quad (1) \]

\(^1\)Overfitting occurs when a statistical model describes random error or noise instead of the underlying relationship

\[ E \sim II + DR + L2M + TT \quad (2) \]

We notice that the execution time (TT) has a strong correlation with energy (since \( energy = power \times time \)). As a result, removing TT from the model deteriorates the performance significantly. Similarly, throughput (TH) (operations per second, e.g GFlops) has a stronger correlation with power compared to the other predictors in the model. Ideally, residuals vs fitted values should demonstrate a random distribution, free of any patterns. Non-linear characteristics are evident from the distribution in Fig. 5. The most common metric to evaluate model performance is R-squared\(^2\) estimate. A pitfall of \( R^2 \) is that the value increases as and when predictors are added to the regression model, and hence, an increase in \( R^2 \) does not necessarily correspond to the performance of the model. We notice a prediction error of over 20% for energy model as shown in Fig. 4 despite a high \( R^2 \) of 0.99. A prediction error of 7% was observed for the power model. This indicates that the models could be tuned further.

**B. Outliers**

Outlying observations are those with extreme values compared to the rest of the data. If there are more outlying observations, the regression model should be investigated for

\[^2\]Coefficient of determination, is defined as the ratio of the sum of squares explained by a regression model and the total sum of squares around the mean

![Fig. 4. Energy and Power estimation using External Test set and 5-fold CV](image)

![Fig. 5. Residuals Vs Fitted for Energy and Power Models (Basic Regression)](image)
errors. Residuals vs leverage plot is one of the many tools available for checking the adequacy of data when fitting regression models. Fig. 6 shows the combined effect of high leverage and regression outliers. Observations #12,#52 (from Fig. 6) could be considered as outlier suspects, as they have extreme values as compared to the rest. Least squares estimates (as the general assumption is that data is normally distributed) for regression models are known to be non-robust to outliers , (for e.g $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \epsilon$) or add polynomial terms to it.

Fig. 6. Residuals Vs Leverage Plot for Energy and Power model (Basic Regression)

We transform explanatory variables (predictors) rather than the response variables (power/energy) as transformation of response variables will affect the relationship of the response variables with all the explanatory variables, not just the one with the non-linear relationship. Apart from transforming certain predictors, another approach would be to try non-parametric regression. We discuss both these options in the ensuing sections.

D. Applying transformations to independent variables

We apply transformations to the independent variables, with an aim to fit the curve to the data points, and then compare the results with basic linear regression model. Analysing the partial residual plots (Fig. 7 and Fig. 8), we get an idea about which predictors fit the current model, and which variables must be transformed in order to improve the fit. After some analysis, we came up with the following transformed power and energy models as follows:

$$ P \sim \frac{II}{DR} + L2M + DR + \frac{1}{TT} + TH^2 \quad (3) $$

Instructions per byte metric has a relation with power (shown in Fig. 1a), represented by $\frac{II}{DR}$ in the formula of the power model.

$$ E \sim TT + II + \log(DR) + \log(L2M) \quad (4) $$

Transformation is generally possible when the relationship between two variables is depicted by a curve that does not change curvature (simple), and the curve is always positive or always negative (monotone). In our case, the relationship pattern between dependent and independent variables cannot be easily modeled using well-known transformations. To improve the models further, we need to look into different modeling strategies. General non-parametric models (of the form $Y = f_1(X_1) + f_2(X_2) + \ldots + f_n(X_n) + \epsilon$) work well with 2-3 predictors (independent variables), but are known to exhibit instability when there are more predictors. Such limitations led us to additive models (which are of the form $Y = f_1(X_1) + f_2(X_2) + \ldots + f_n(X_n) + \epsilon$), to apply local regression using functions that are automatically estimated from the data.

Fig. 7. Partial Residual plot of Power

Fig. 8. Partial Residual plot of Energy
E. Generalized Additive Models (GAM)

Instead of trying to fit a single parameter in a model (that represents relationship between independent and dependent variables), GAM performs an operation called "smoothing" on independent variables, to estimate power/energy. The idea of GAM is to fit a model of the form:

\[ Y = f_0 + \sum_{j=1}^{p} f_j(X_j) + \epsilon \]

where \( \epsilon \) has mean 0 and finite variance, \( X_j \) represents variables or functions of independent variables, and the functions \( f_j(\ldots) \) are the smoothers. The `gam` function of mgcv library of \( \mathbb{R} \) statistical package was used to perform all the tests. Using GAM, Fig. [10] shows an improvement in power and energy prediction on an average of \( \sim 50\% \) and \( \sim 80\% \) respectively compared to the initial regression model (shown in Fig. [4]). We use the same model formula (as in Eq. [3] and Eq. [4]) for the transformed power/energy models, but apply GAM automatic "smoothers" on the predictors.

### V. Comparing the Performance of Models

In this section, we evaluate the models described in the previous section, and compare their performances. Regression diagnostics is a diverse topic, however, we have primarily depended on residual versus fitted, normality test and residuals versus leverage plots to evaluate a model. Our ultimate aim is to minimize the prediction errors by applying different statistical tools. Table II lists the improvements in average prediction error for power model.

![Fig. 9. Energy and Power estimation with transformations using External Test set and 5-fold CV](image)

![Fig. 10. Power and Energy estimation using Generalized Additive Model](image)

#### TABLE II

<table>
<thead>
<tr>
<th>Versions</th>
<th>Average Prediction Error %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Linear Regression</td>
<td>7</td>
</tr>
<tr>
<td>Linear Regression With Transformations</td>
<td>3.62</td>
</tr>
<tr>
<td>GAM</td>
<td>3.5</td>
</tr>
</tbody>
</table>

Similarly, Table III shows the improvements in the prediction error for energy estimation.

#### TABLE III

<table>
<thead>
<tr>
<th>Versions</th>
<th>Average Prediction Error %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Linear Regression</td>
<td>23.49</td>
</tr>
<tr>
<td>Linear Regression With Transformations</td>
<td>12.5</td>
</tr>
<tr>
<td>GAM</td>
<td>4</td>
</tr>
</tbody>
</table>

The \( R^2 \) estimate on its own does not indicate the correctness of a model, we need to explore the residual distribution and check the prediction error that a model produces. Instead, we look into the Q-Q plots of the models, which is a better tool to explore the behavior of predictive models. A normal Q-Q (Quantile-Quantile) plot of regression is another popular diagnostic tool for checking whether the residuals are normally distributed. If the data follows the distribution, then points on the plot would fall very close to the straight line. Fig. [11c] and Fig. [11d] show an improvement of the transformed model (more points closer to the line), using the Q-Q plots. The GAM model shown in Fig. [11c] and Fig. [11d] is slightly better than the transformed linear regression.

It could be observed from Fig. [11] and Table III that by just tuning the model we could improve the prediction performance by \( \sim 80\% \).

### VI. Related Work

Hong and Kim [3] proposed an integrated power model to simulate GPU performance with power and temperature.

\( ^3 \)a statistical technique for estimating a real valued function which represents a smooth generalization of two variables

\( ^4 \)It plots quantiles of the data versus quantiles of a distribution
They use a timing model and estimate hardware access rates required by their model. Whereas, we rely on few hardware counters (along with some performance attributes easily obtainable from the application) that are critical to investigate the performance of an application on a GPU, irrespective of GPU architecture/generation. In [4], the authors used Support Vector Regression analysis (SVR) aiming to augment regression models, and show significant differences in prediction accuracies across regression models, stressing the importance of statistical tools in fine-tuning a model.

VII. CONCLUSION AND FUTURE WORK

We have shown that it is possible to estimate power and energy with a small set of independent variables, in the form of hardware counters and performance attributes. Our average errors for predicted values of power and energy are well within 4%. However, statistical tests rely on plenty of input data, and although we sampled data from multiple runs of 4 different kernels, it is still not enough to accurately estimate a vast range of possibilities. In near future, we plan to include more data from many applications, and analyze using different regression techniques.

VIII. ACKNOWLEDGMENT

Development at the University of Houston was supported in part by NSF CCF-0917285 and CRI-0958464. Thanks to Priyanka Ghosh for her valuable comments.

REFERENCES


---

5 A linear regression technique which penalizes the size of regression coefficients.