A MEASUREMENT STUDY OF GPU DVFS ON ENERGY CONSERVATION

Xinxin Mei, Ling Sing Yung, Kaiyong Zhao and Xiaowen Chu
Department of Computer Science
Hong Kong Baptist University

5th Workshop on Power-Aware Computing and Systems
3, November, 2013
Outline

- Introduction
- Experimental Methodology
- Experimental Results
- Conclusions
INTRODUCTION

- Background & motivation
- Objective & evaluation
Background

- GPU DVFS (dynamic voltage/frequency scaling)
  - Simulation:
    Many core sub-45 nm predictive technology model
  - Measurement:
    Frequency scaling only

We are going to observe real DVFS impact by scaling both voltage and frequency.
Objective

How much **system energy** could be saved by DVFS?

- Core voltage ($V_{\text{core}}$);
- Memory/DRAM voltage ($V_{\text{mem}}$);
- Core frequency ($f_{\text{core}}$);
- Memory (DRAM I/O) frequency ($f_{\text{mem}}$)

$$E = t\overline{P_i}$$

An example of system power consumption when GPU executes a program.
Evaluation

Energy savings:

\[
\hat{R} = 1 - \frac{E_{\text{min}}}{\hat{E}}
\]

compared to factory default setting

\[
R_{\text{max}} = 1 - \frac{E_{\text{min}}}{E_{\text{max}}}
\]

the maximum saving ability
METHODOLOGY

- Tools
- Benchmarks
- Platform
Tools

- **Scaling Tool**
  - MSI Afterburner
  - NVIDIA Inspector

- **Energy measurement**
  - Watts up? PRO

The same voltage/frequency setting during program execution
Benchmarks

- 37 programs: Rodinia 2.1 & CUDA SDK 4.1

- Revise data size & kernel iteration time
  - Data size: as large as possible
  - Iteration time: execution time > 30 seconds
Platform

- **MSI N560GTX-Ti Hawk**

<table>
<thead>
<tr>
<th>Category</th>
<th>Default</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{\text{core}}$ (V)</td>
<td>1.049</td>
<td>[0.849, 1.149]</td>
</tr>
<tr>
<td>$f_{\text{core}}$ (MHz)</td>
<td>950</td>
<td>[480, 1000]</td>
</tr>
<tr>
<td>$V_{\text{mem}}$ (V)</td>
<td>1.50</td>
<td>[1.40, 1.58]</td>
</tr>
<tr>
<td>$f_{\text{mem}}$ (MHz)</td>
<td>2100</td>
<td>[1050, 2300]</td>
</tr>
</tbody>
</table>

- **NVIDIA GTX 560 Ti**

<table>
<thead>
<tr>
<th>Category</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{\text{core}}$ (V)</td>
<td></td>
</tr>
<tr>
<td>$f_{\text{core}}$ (MHz)</td>
<td>822</td>
</tr>
<tr>
<td>$V_{\text{mem}}$ (V)</td>
<td></td>
</tr>
<tr>
<td>$f_{\text{mem}}$ (MHz)</td>
<td>2004</td>
</tr>
</tbody>
</table>
RESULTS

- Core scaling
- Memory scaling
Core Scaling

- Nearly all programs: higher core frequency, less energy
- Best compute capacity space: low core voltage & high core frequency at the same time
V_core/ f_core Scaling Efficiency

Average: $\bar{R} = 19.28\%$; $R_{\text{max}} = 24.40\%$

Most beneficial programs: both memory & computation intensive
Core Scaling

- Low core voltage conserves energy

![Bar chart showing energy savings and performance decrease](chart.png)
Chip Temperature

- **Lower** core voltage, **lower** temperature increasing rate
Memory Scaling

- Memory voltage almost has **no influence**
  narrow range & small power contribution
F_mem Scaling

Average: $\hat{R} = 3.52\%$; $R_{max} = 10.20\%$

Most beneficial programs: low memory & computation parallelism
Execution time

$f_{\text{mem}}$ : influence energy by influencing **execution time**

![Graph showing normalized execution time vs. memory frequency]
CONCLUSION
Conclusion

Voltage/frequency scaling can save energy!

- Scaling core voltage & core frequency
  - **Low** core voltage, always
  - **High** core frequency, most of the time

- Scaling memory frequency
  Optimal memory frequency depends on program characteristics
Q1: Do we consider the power phase phenomenon?

No, we just take the average of power samples. We failed to observe phase phenomenon of some of our benchmarks due to the execution time limitation. On the other hand, our power meter takes a sample at a relatively large interval. It cannot show real-time power consumptions as those of simulators.
mailto: xxmei@comp.hkbu.edu.hk