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:

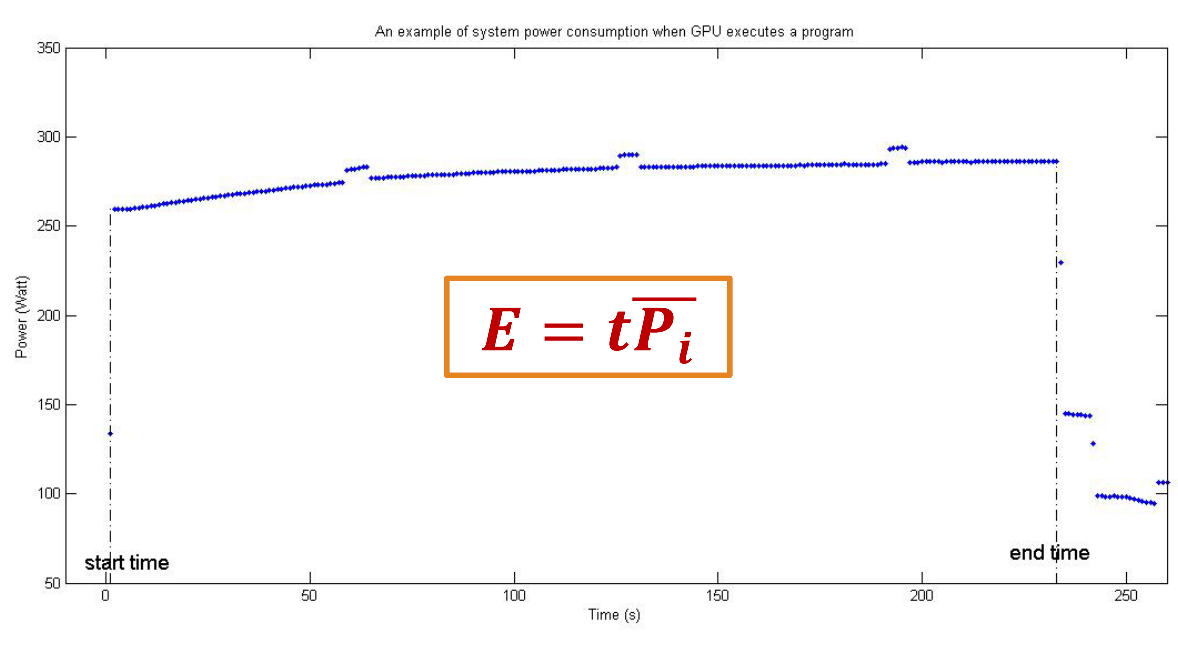
    Frequency scaling only

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



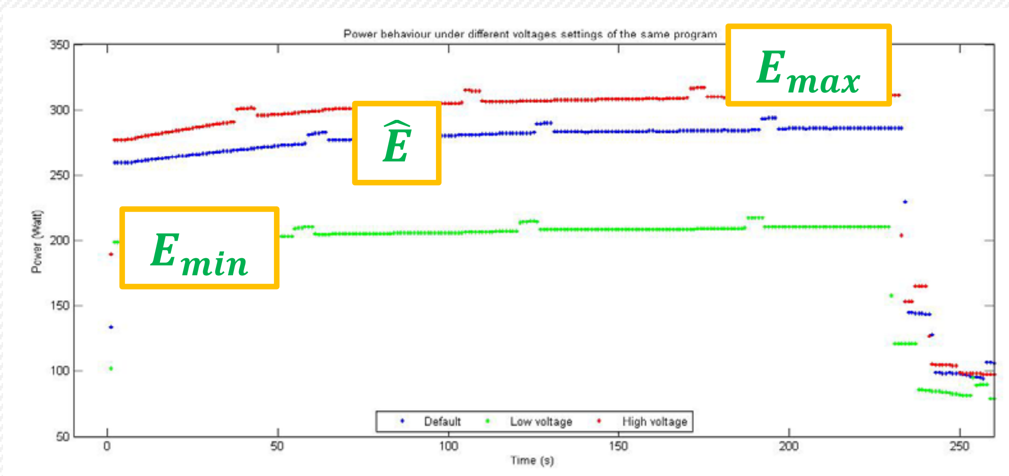
$$E = t\overline{P_i}$$

Evaluation

Energy savings:

$$\hat{R} = 1 - \frac{E_{min}}{\hat{E}}$$ compared to factory default setting

$$R_{max} = 1 - \frac{E_{min}}{E_{max}}$$ the maximum saving ability

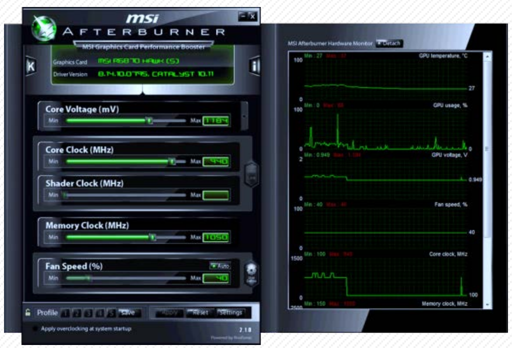


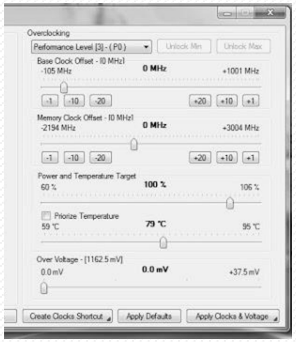
METHODOLOGY

- 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

Category	Default	Range
V_core (V)	1.049	[0.849, 1.149]
$\boldsymbol{f\_core}$(MHz)	950	[480, 1000]
V_mem (V)	1.50	[1.40, 1.58]
f_mem (MHz)	2100	[1050, 2300]

- NIVIDIA GTX 560 Ti

Category	Default
V_core (V)	
$\boldsymbol{f\_core}$(MHz)	822
V_mem (V)	
f_mem (MHz)	2004

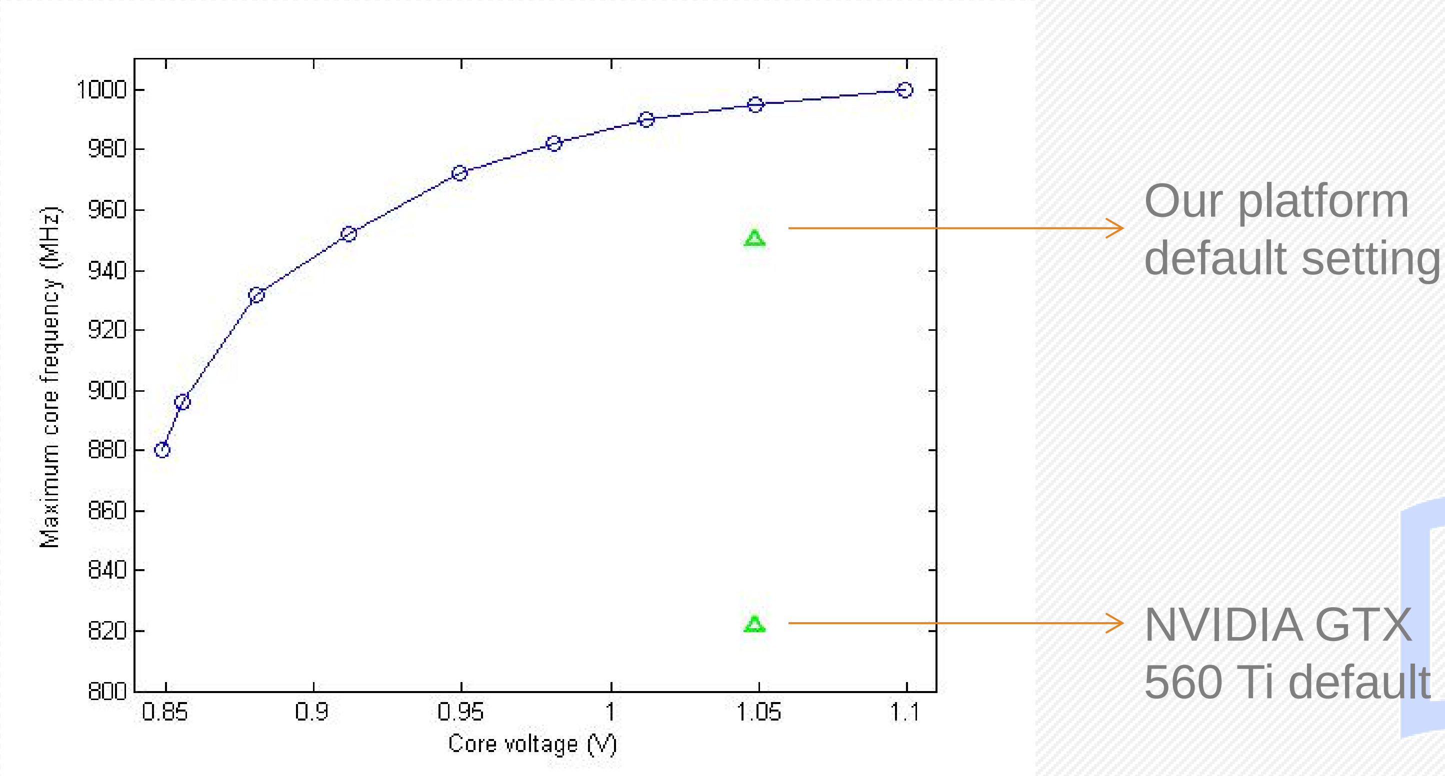
RESULTS

- Core scaling
- Memory scaling

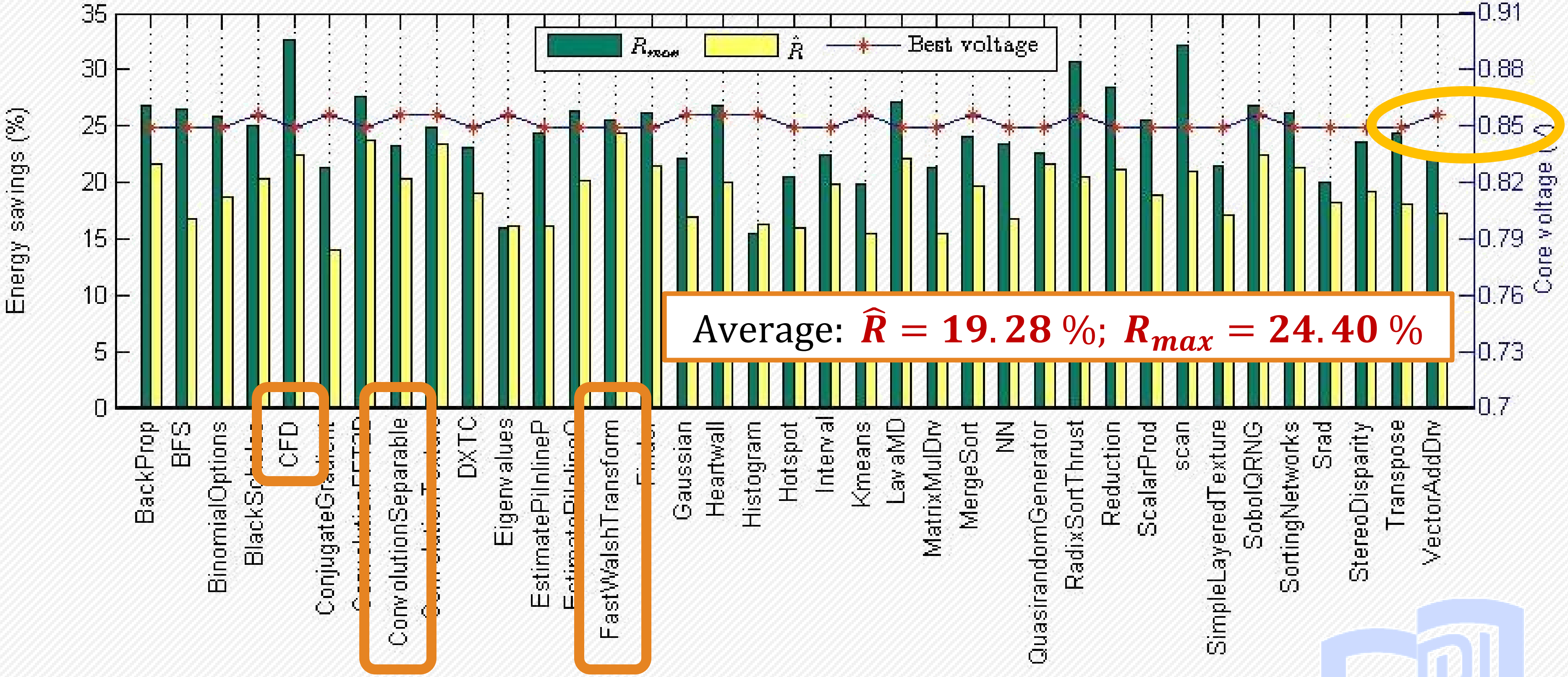
Core Scaling

- **Nearly** all programs: **higher core frequency, less energy**
- Best compute capacity space:

  low core voltage & high core frequency at the same time



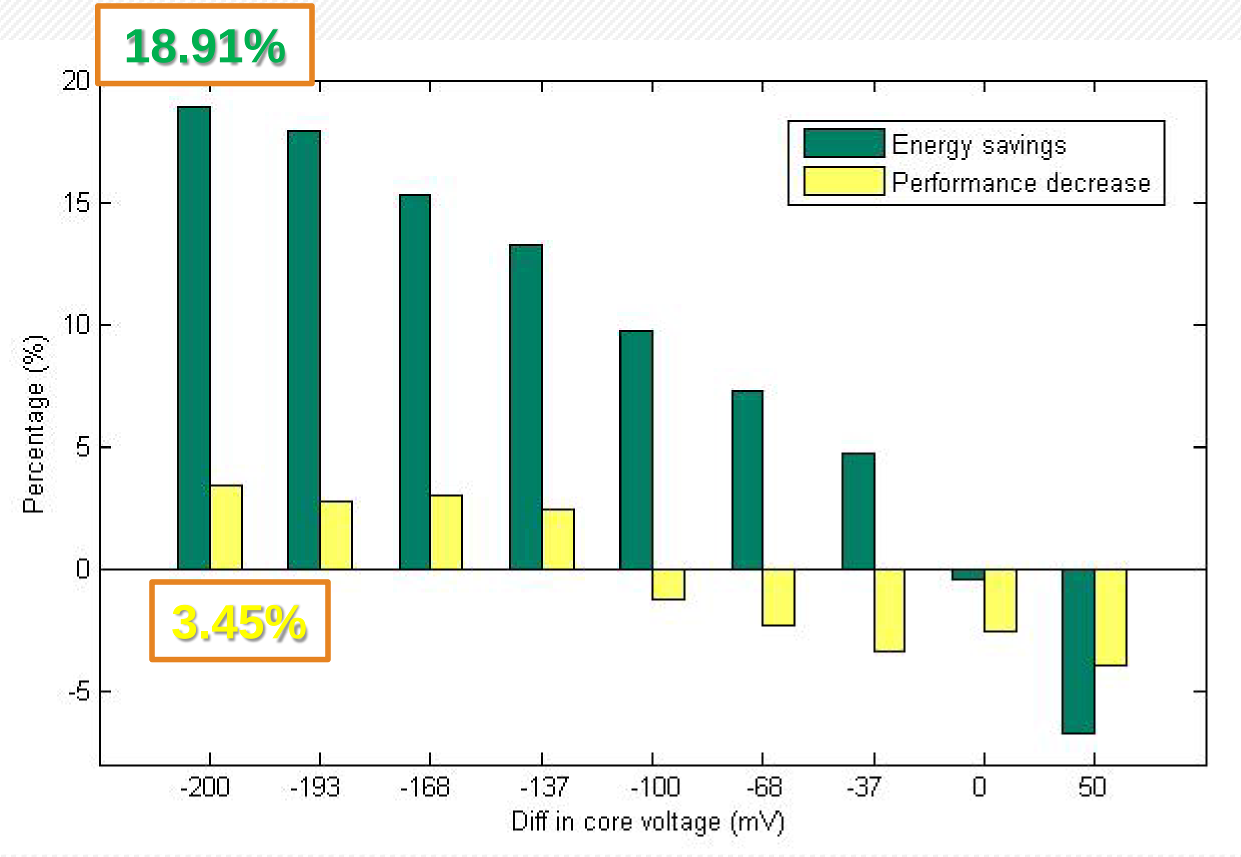
V_core/ f_core Scaling Efficiency



Most beneficial programs: **both memory & computation intensive**

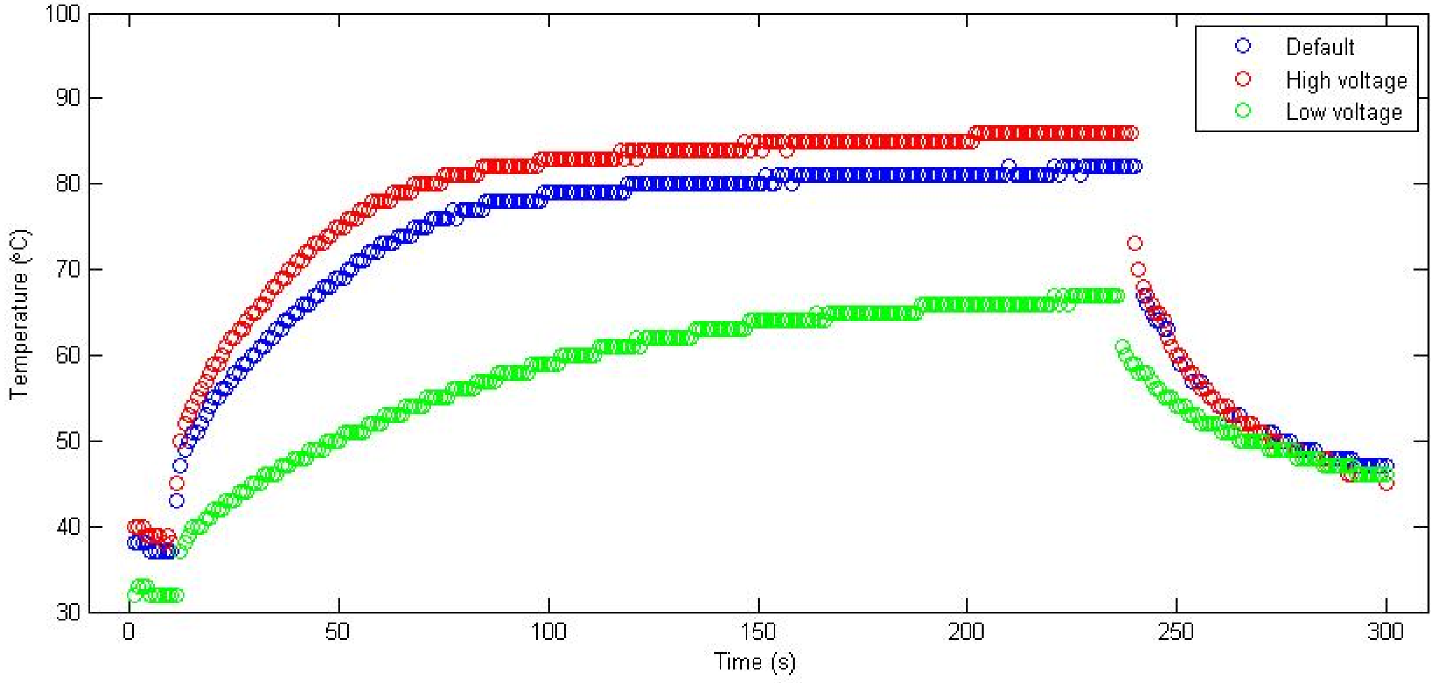
Core Scaling

- Low core voltage conserves energy



Chip Temperature

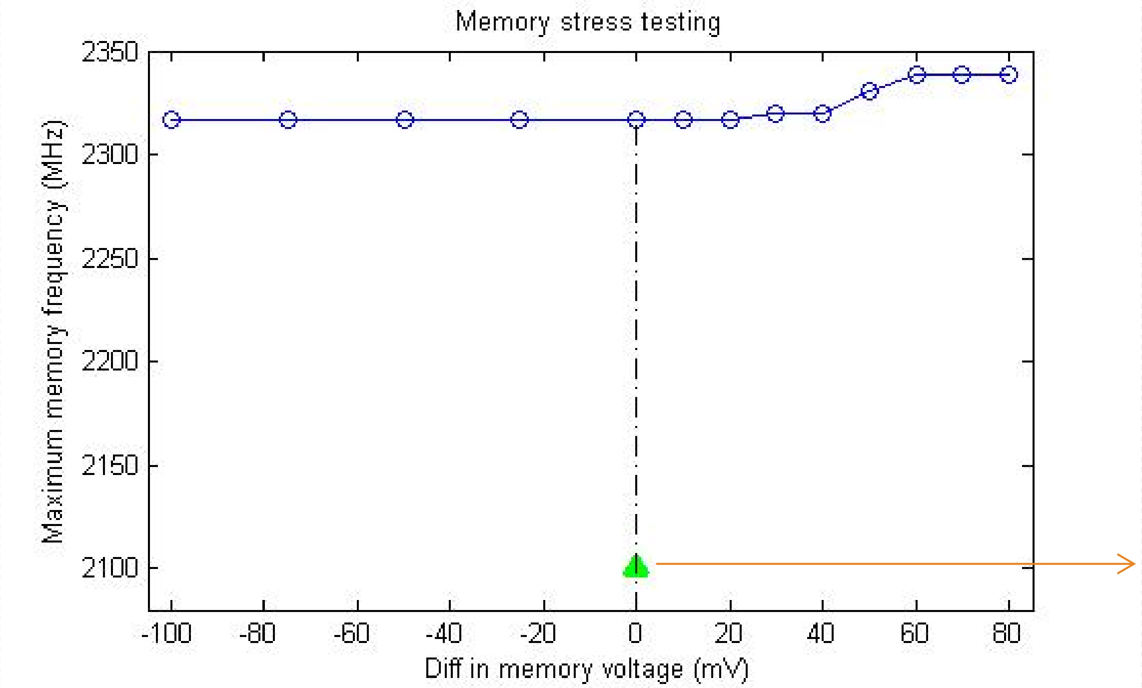
- **Lower** core voltage, **lower temperature increasing rate**



Memory Scaling

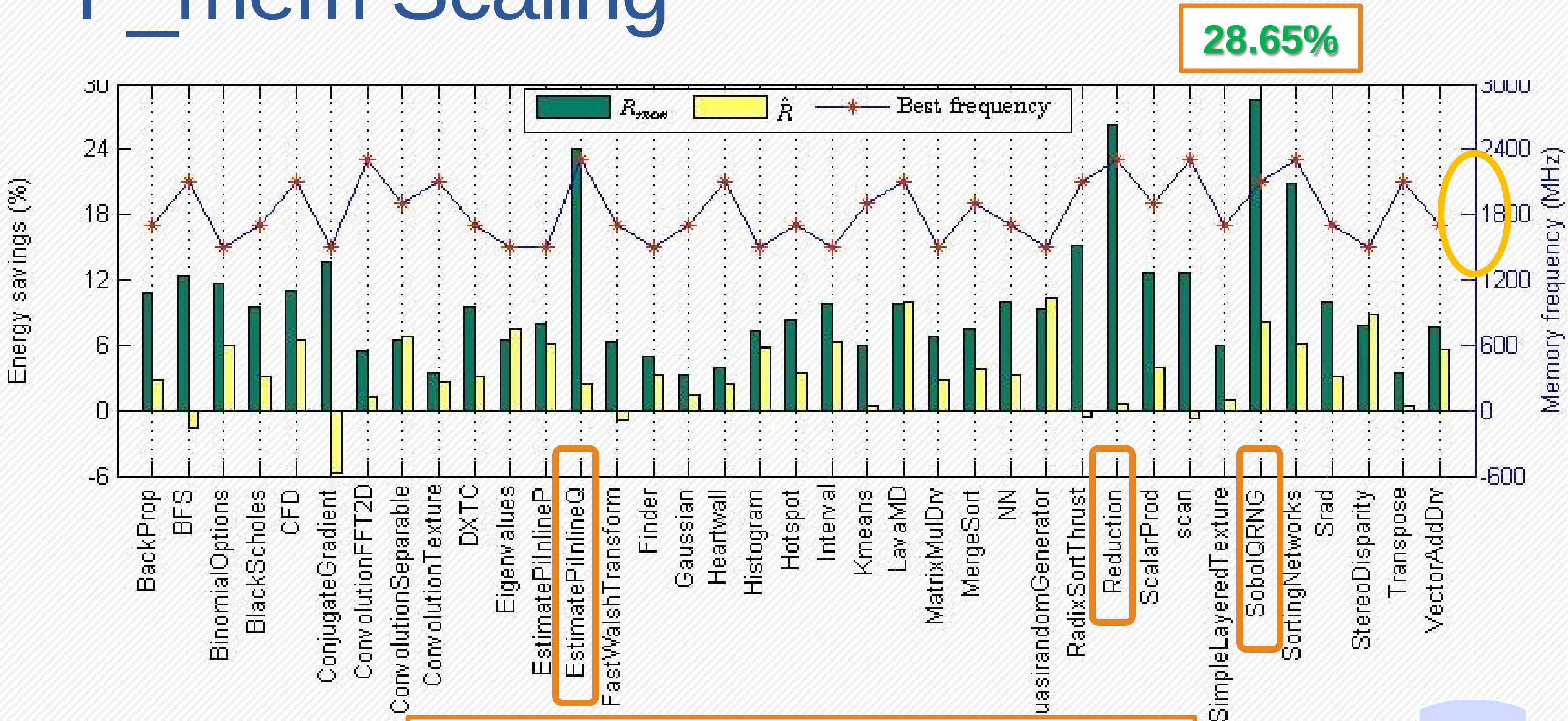
- Memory voltage almost has **no influence**

  narrow range & small power contribution



Our platform default setting

F_mem Scaling

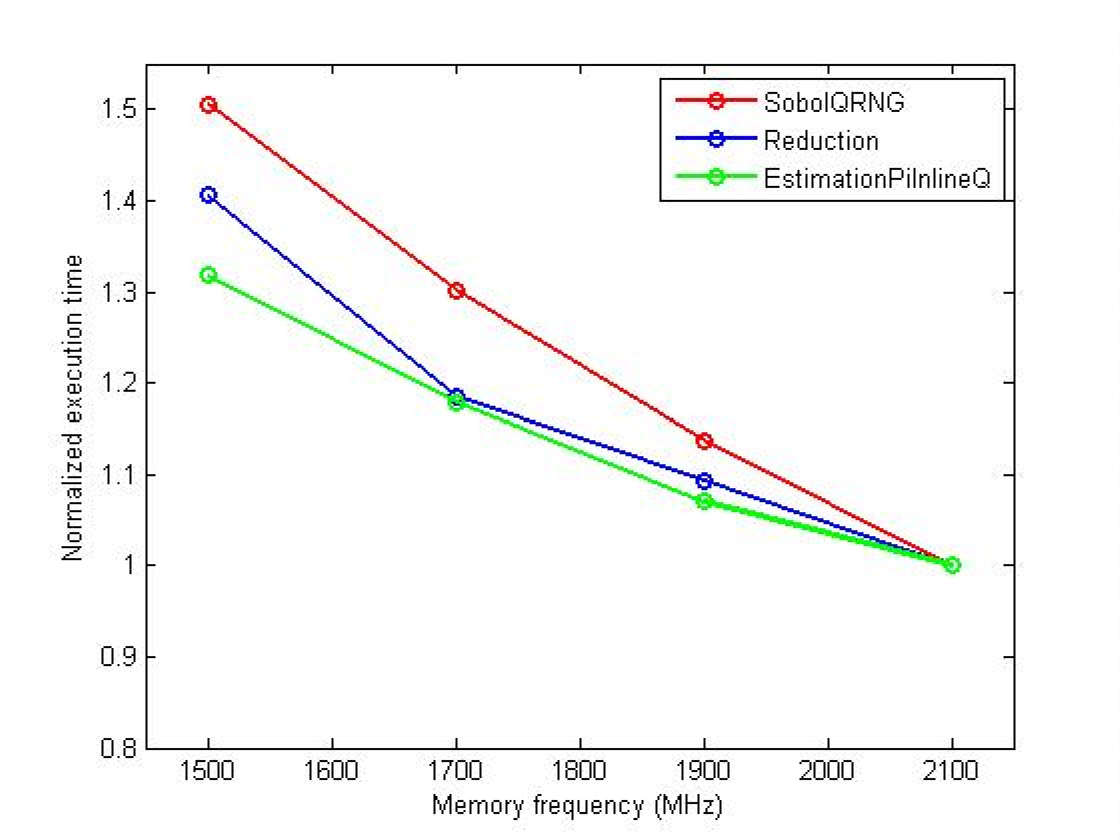


Average: $\hat{R} = 3.52\ \%$; $R_{max} = 10.20\ \%$

Most beneficial programs: **low memory & computation parallelism**

Execution time

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



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


Q&A

Supplemental Slides

- 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 our benchmarks due to the execution time limitation. On the other hand, our power meter takes an sample at a relatively large interval. It can not show real time power consumptions as those of simulators.

mailto: xxmei@comp.hkbu.edu.hk


Thanks!