Statistical power modeling of GPU kernels using performance counters

Conference Paper · August 2010

DOI: 10.1109/GREENCOMP.2010.5598315 · Source: DBLP

Naoya Maruyama
RIKEN

Toshio Endo
Tokyo Institute of Technology

Statistical Power Modeling of GPU Kernels Using Performance Counters

Hitoshi Nagasaka, *Naoya Maruyama*, Akira Nukada, Toshio Endo and Satoshi Matsuoka

Tokyo Institute of Technolog

International Green Computing Conference, Aug 2010

GPGPU in Scientific Computing

- Strong performance advantages in throughput-oriented scientific computing
  - Tesla S2050: 515 GFlops (DP), 150 GB/s
- Getting integrated in large-scale supercomputing sites
  - Tokyo Tech TSUBAME
  - China’s Nebulae
  - ORNL
  - NCSA/UIUC

Power Consumption by GPU

- TDP 250 W per board (Geforce GTX 480)
  - Twice as the typical CPU socket's consumption
- Little is known about fine-grained characteristics
  - How much is it affected by *actual workloads*?
  - Is it more *power efficient* than using the CPU?
- Should be able to know power consumption in a *program-by-program basis* and *accurately*

Objective

- Allow the power consumption of each GPGPU program to be known accurately
  - Should be easily applicable to real non-trivial programs

Approach

- Statistical estimation with performance counters
  - Known to be effective in the CPU
  - Not yet studied in the GPU

Contributions

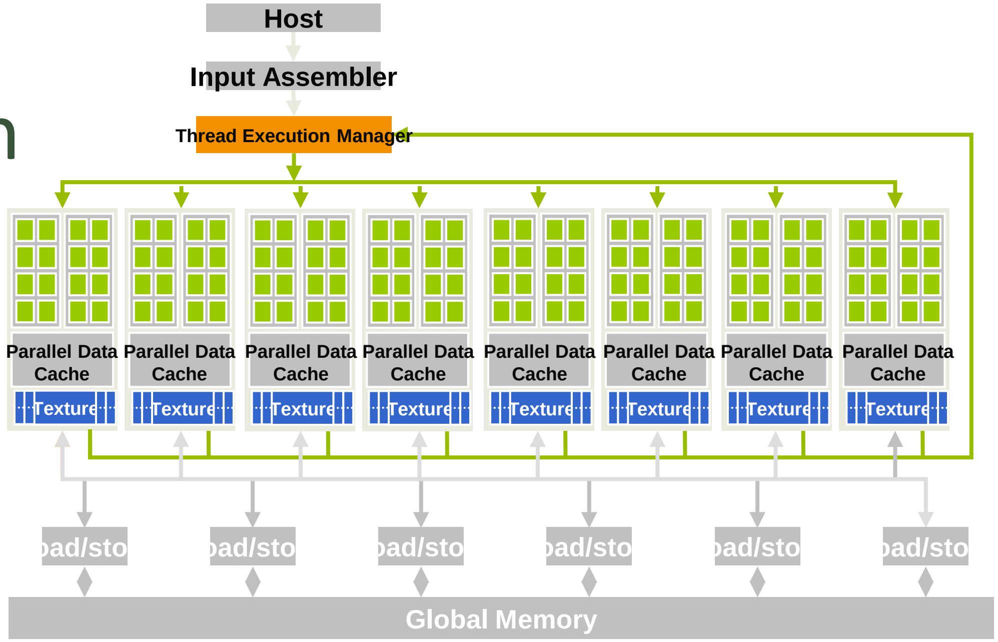
- Demonstrates effectiveness of PMC-based energy modeling in NVIDIA CUDA GPUs
  - 95% accuracy in average
- Insights in how the GPU consumers power
  - Linearly correlated with instruction and memory throughputs

Outline

1. Introduction
2. GPGPU Overview
3. Statistical Modeling
4. Experimental Results
5. Related Work & Conclusions

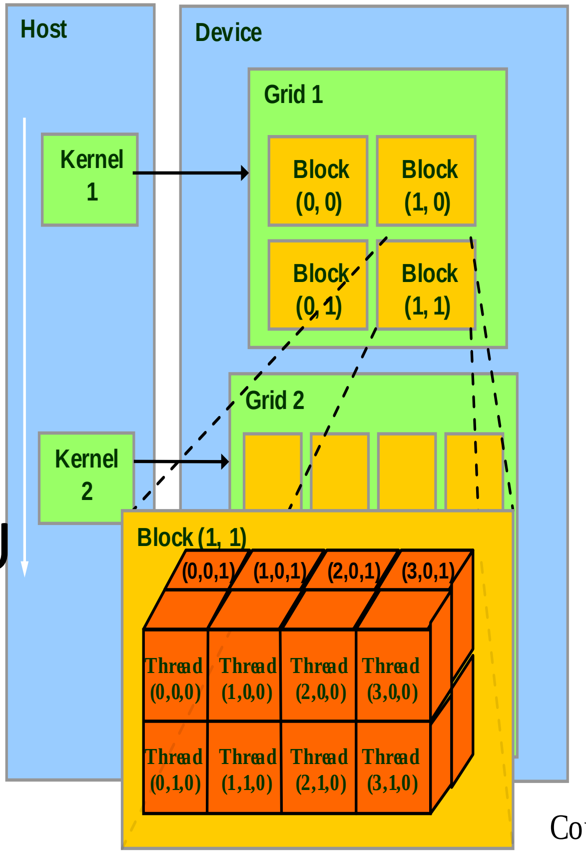
Overview of NVIDIA GPU

- Currently the dominant HPC accelerator
- Consists of multiple 32-wide SIMD cores
  - The number of cores depend on GPU models and architectures
  - A huge number of simultaneous threads by the HW scheduler
- Coupled with DRAM on GPU boards
  - Hundreds of MB to a few GB



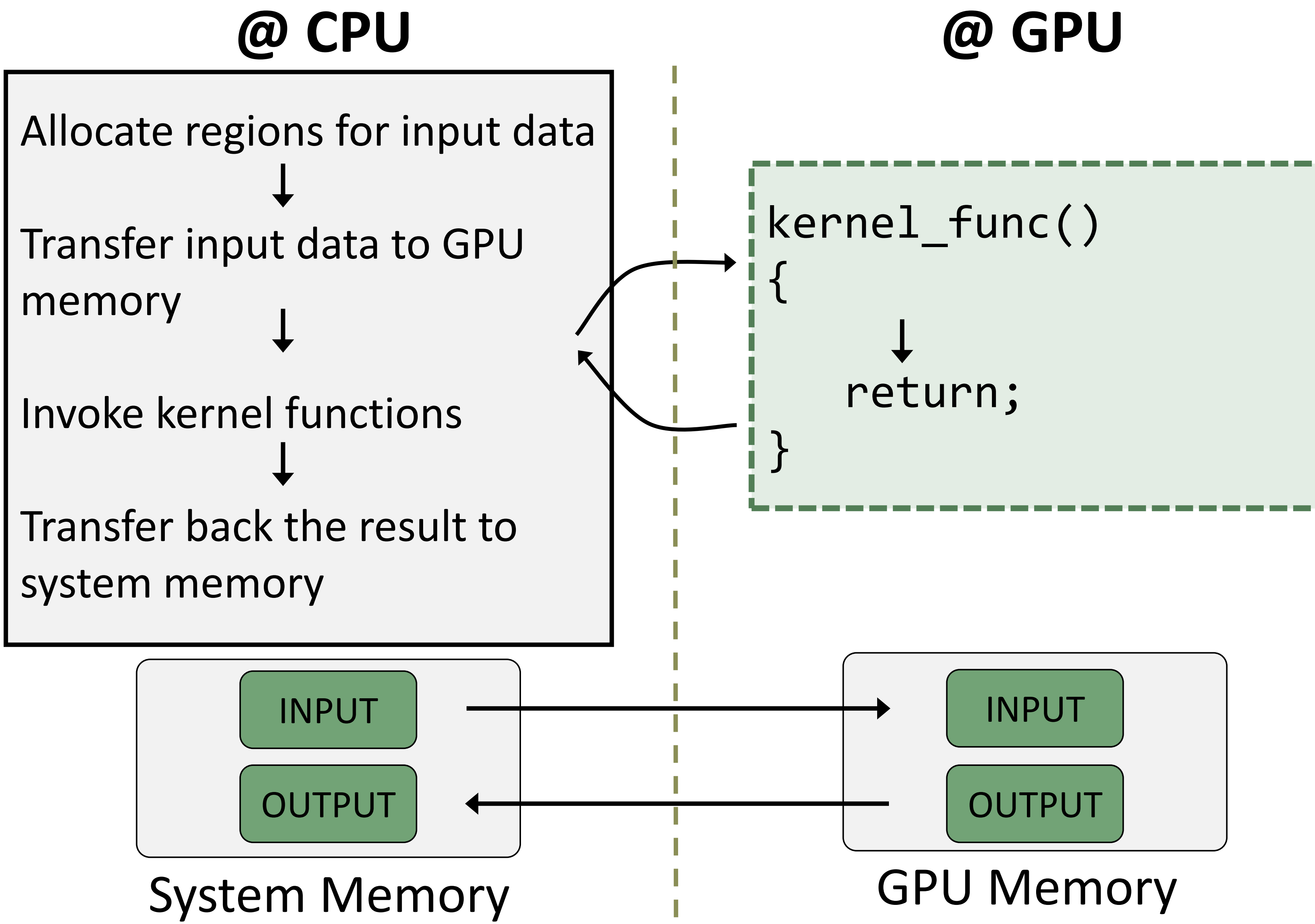
Programming the GPU with CUDA

- Provides abstractions for programming the NVIDIA GPU
  - *Threads, thread blocks, grids*
  - *Global/texture/shared/local memory*
- Consists of *host* and *kernel* functions
  - Host: functions running on the CPU to control the GPU
  - Kernel: functions running on the GPU in parallel



Courtesy: NDVI

Typical Execution Flow



GPU Energy Modeling

- Motivation
  - No standard ways for direct measurements
  - Needs alternative methods based on *measurable* metrics
- Approach: Statistical modeling based on performance profiles
  - Statistically correlate **performance** and **power**
  - Past work: PMC-based modeling on the CPU
  - NVIDIA GPUs are also equipped with several PMCs

Measuring GPU Power Consumption

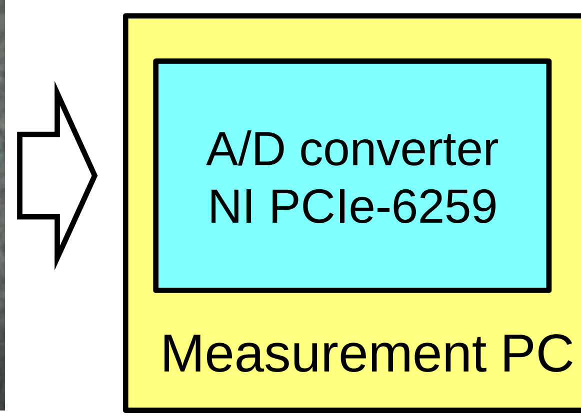
- Two power sources
  - Via PCI Express: < 75 W
  - Direct inputs from PSU: < 240 W
- Uses current clamp sensors around the power inputs

Precise and Accurate Measurement with Current Probes

Attaches current sensors to two power lines in PCIe

+

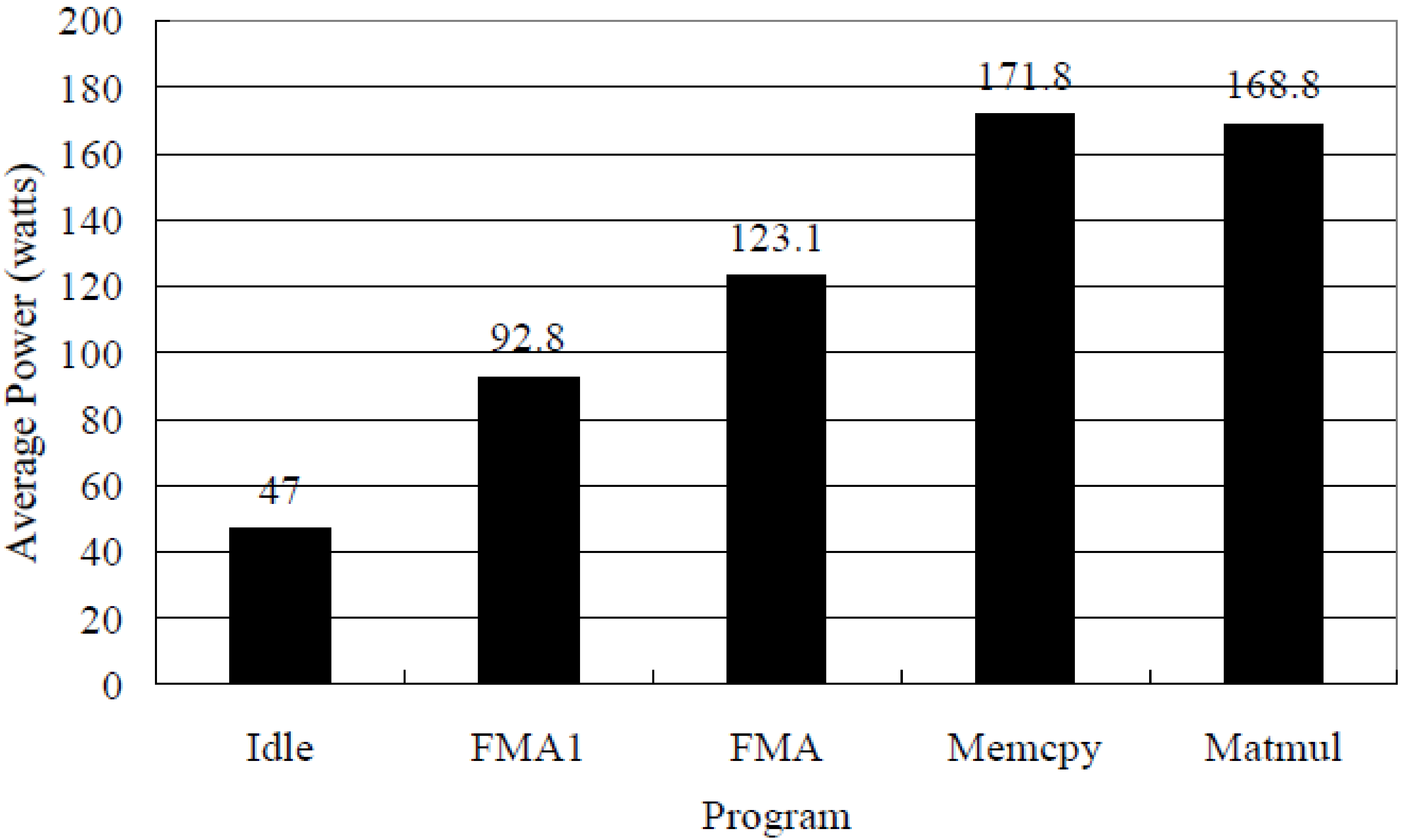
Direct power inputs from PSU



Reads currents at 100 us interval

Power Profile of GeForce GTX 285

TDP: 200 W



Performance Profiling in GPU

- Obtaining PMC values for each kernel execution via the CUDA Profiler

Name	Description
gld_32b	32-byte global memory load transactions
gld_64b	64-byte global memory load transactions
gld_128b	128-byte global memory load transactions
gst_32b	32-byte global memory store transactions
gst_64b	64-byte global memory store transactions
gst_128b	128-byte global memory store transactions
local_load	Local memory loads
local_store	Local memory stores
branch	Number of branches
divergent_branch	Number of divergent branches
instructions	Instructions executed
warp_serialize	Number of thread warps that has bank conflicts
tlb_miss	Number of TLB misses

Statistical Modeling

- Regularized linear regression
  - Finds linear correlation between per-second PMC values and average power of a kernel execution
  - Aggregates 3 kinds of global memory loads/stores
    - gld: gld_32b + gld_64b * 2 + gld_128b * 4
  - Regularization for avoiding overfitting to training data (Ridge Regression [10])

Average power of a kernel (Watts)

$$p = \sum_{i=1}^{n} \alpha_i c_i + \beta$$

Per-second PMC values

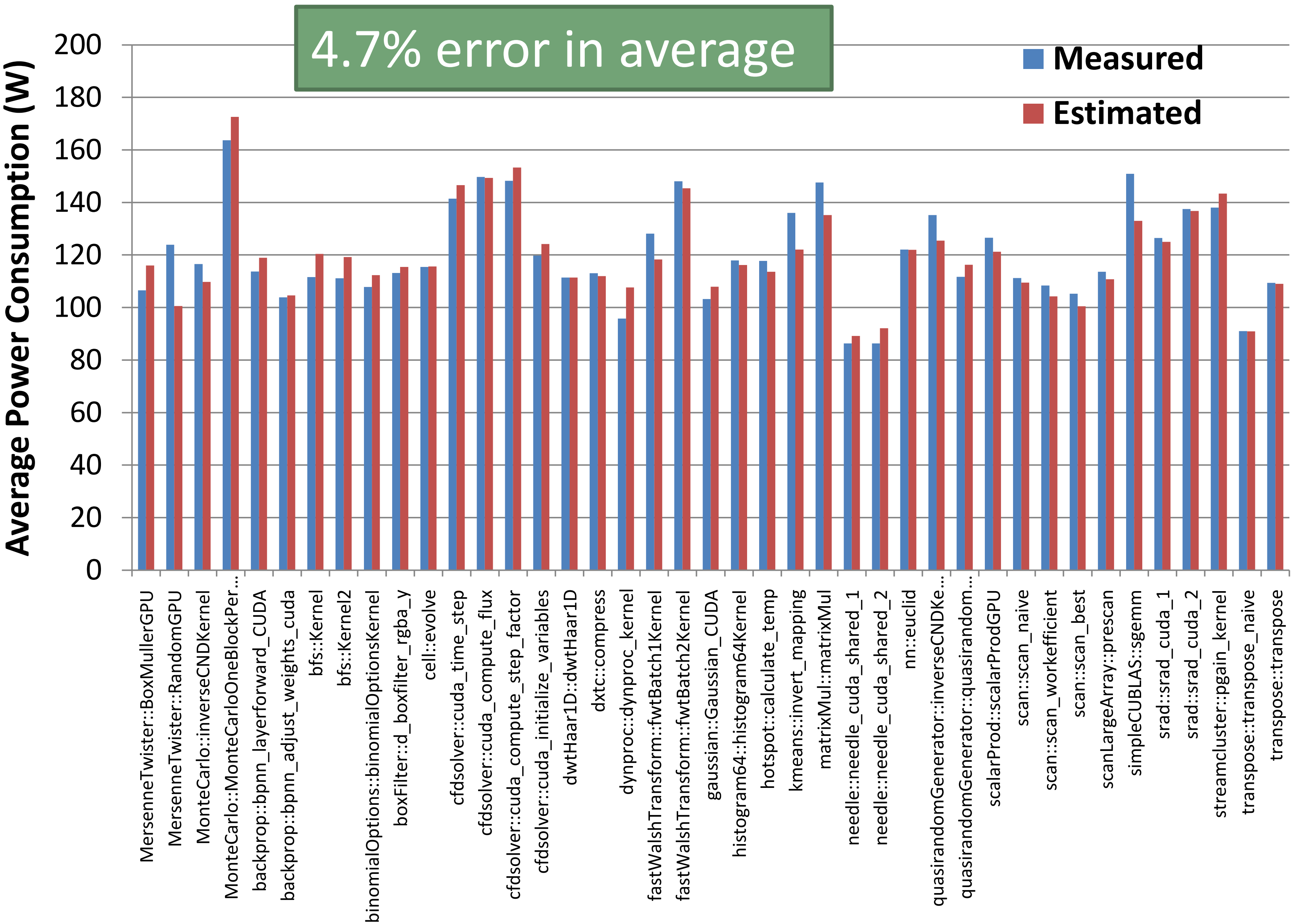
Evaluation

- Tests the accuracy of the proposed modeling
- Methodology
  - Use publicly available CUDA programs to derive and test the model
  - 5-fold cross validation
    - Training using 4/5 of data sets
    - Evaluates accuracies by applying the model to 1/5 data
- Machine setup
  - NVIDIA GeForce GTX 285 (a high-end model in the last generation NVIDIA GPUs)
  - AMD Phenom 9850 Quad-Core processor, AMD 790FX, 4GB of DDR3 DRAM
  - 64-bit Linux with gcc 4.3.2 and CUDA 2.3

Collecting Sample Data

- 43 GPU kernels
  - 17 applications from NVIDIA CUDA SDK
  - 13 applications from the Rodinia Benchmark
  - Prunes the kernels with less than 120 thread blocks to minimize the effect of using skewed PMC values
  - Excludes kernels involving texture reads
- Run each kernel multiple times for 1 second for power measurement
  - Compensates time skews between the GPU machine and the measurement machine
  - Not necessary once the model is derived

Actual vs. Estimated Power

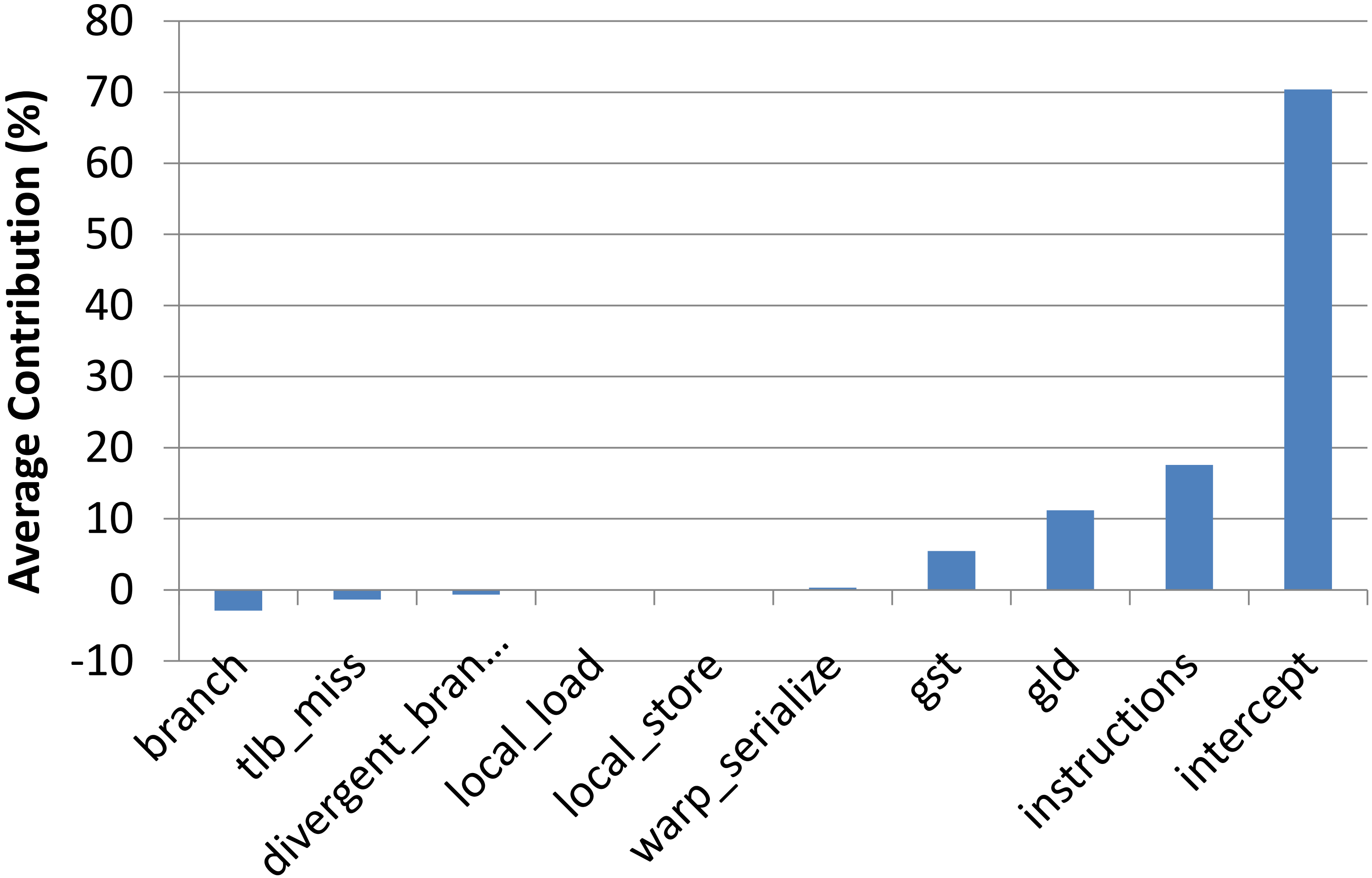


Sources of Large Errors

- 23W lower than the actual with kernel RandomGPU in program MersenneTwister
  - Likely to be caused by insufficient training data
  - The kernel is the only one that has `local_load` and `local_store`
- 18W lower than the actual with kernel `sgemm` in program simpleCUBLAS
  - SGEMM is likely to exercise FP units more extensively
  - Such differences are not reflected in performance profiles since there is no PMC for FP ops

Linear Model Contributions

- Variations are dominated by instruction and memory throughputs



Limitations

- Texture reads
  - Texture hardware can improve latencies in irregular GPU DRAM accesses
  - But, no PMC as of this study (available in more recent GPUs)
  - Resulted in large errors when applied to such kernels
    - Actual: 150 W vs. Estimated: 100 W (boxFilter)
- Multiple training runs
  - 4 runs to cover the 13 PMCs
  - #instructions and #accesses might be enough

Related Work

- Much work on statistical modeling of power consumption for the CPU
  - [Isci and Martonosi, 2003], [Bellosa et al., 2003], [Lee and Brooks, 2006], etc.
- Analysis and modeling of power consumption of graphics performance profiles [Ma et al., 2009]
  - Does not capture GPGPU-specific events
  - This study: More accurate results with more thorough experiments

Conclusions

- Summary
  - Accurate statistical modeling of GPU power consumption
  - Once a power model is derived, no need for direct measurements
- Future work
  - Evaluates the effectiveness of the proposed methodology in different GPU architectures (NVIDIA Fermi, AMD, etc)
  - Energy optimization with the model
    - Clock scaling based on estimated power efficiency

Extra slides

Limitations in GPU Profiling

- Only 4 HW PMC registers available
  - Needs to run the same kernel multiple times to collect more than 4 counter values
- PMCs are only read from one SIMD core (SM)
  - Executions with load imbalance between SIMD cores can yield skewed profiles
  - Uses only kernels with at least a certain number of thread blocks (120 in our experiments)
- Texture reads are not monitored
  - Texture reads involve DRAM accesses
  - Can be a source of modeling errors
  - Solved in the recent GPUs

Texture Reads

- Some kernels use texture hardware for accessing DRAM data
  - Improves latencies in irregular data accesses
  - But, no PMC as of this study (available in more recent GPUs)
- Evaluates accuracies in two cases
  - Case 1: kernels without texture reads (41 kernels)
  - Case 2: all kernels (Case 1 + 6 additional kernels)

Kernels incl. Texture Reads