Overview of Performance Prediction Tools for Better Development and Tuning Support

Universidade Federal Fluminense

Rommel Anatoli Quintanilla Cruz / Master's Student
Esteban Clua / Associate Professor

GTC 2016, San Jose, CA, USA, April 7th, 2016
What you will learn from this talk ...
Outline

• Motivation
• Performance models
• Applications
• Challenges
Performance Optimization Cycle*

1. Profile Application
2. Identify Performance Limiters
3. Analyze Profile & Find Indicators
4. Reflect
5. Change and Test Code

* Adapted from S5173 CUDA Optimization with NVIDIA NSIGHT ECLIPSE Edition – GTC 2015
Performance Analysis Tools

NVIDIA Visual Profiler

The NVIDIA CUDA Profiling Tools Interface

The PAPI CUDA Component
Performance tools are still evolving

CUDA 7.5 Instruction-level profiling

Synchronization stalls reduced, memory dependency stalls are removed

NVIDIA Visual Profiler
Performance tools are still evolving

But it's still not enough ...

Concurrent Kernel Execution

Power

Streaming
Outline

• Motivation
• Performance models
• Applications
• Challenges
Performance models
Performance models

- Pseudocode
- Source code
- PTX
- CUBIN
- Target Device Information

Input
Performance models

**Input**
- Pseudocode
- Source code
- PTX
- CUBIN
- Target Device Information

**Performance model**
- Power consumption estimation
- Execution time prediction on a target device
- Performance bottlenecks identification

**Output**
Types of performance models

Analytical Models

Statistical Models

Simulation

Advantages & Disadvantages
The MWP-CWP model [Hong & Kim 2009]

MWP: Memory warp parallelism
CWP: Computation warp parallelism
Statistical models

Simulation

GPU Ocelot

PTX Kernel

PTX Emulation

GPU Execution

LLVM Translation
Outline

• Motivation
• Performance models
• Applications
• Challenges
Applications of performance models

Successfully used to …

• schedule concurrent kernels
• make auto-tuning
• estimate power consumption
• identify performance bottlenecks
• make workload balancing

Today!
Auto-tuning

- Optimization goals
- Parameters
- Large search space
Concurrent Kernel Execution

Supported since Fermi

Limitations: Registers, Shared Memory, Occupancy

* Image from http://www.turkpaylasim.com/cevahir
Outline

• Motivation
• Performance models
• Applications
• Challenges
Challenges

- Multiple-gpu systems, heterogeneous systems
- Each microarchitecture has its own features
- More complex execution behavior is harder to model accurately
References and Further reading


Acknowledgements
Thank you!

JOIN THE CONVERSATION
#GTC16

Contact:
ruquintanillac@ic.uff.br
esteban@ic.uff.br

http://medialab.ic.uff.br
Questions & Answers
Backup Slides
Simplified compilation flow

1. **Host code**: `.cpu`
2. **Device code**: `.gpu`
3. **Virtual Instruction Set**: `.ptx`
4. **CUDA Binary File**: `.cubin`
5. **CUDA Executable**: `.fatbinary`

**CUDA Compiler**
- `nvcc`

**CUDA Front End**
- `cudafe`
- `ptxas`
- `ptx`

**Host Compiler**
- `nvcc`
- `cicc`

**PTX Optimizing Assembler**
- `ptxas`

**High level optimizer and PTX generator**
- `cicc`

**CUDA Executable**
Concurrent Kernel Execution

Leftover policy

Timeline

K1 16 blocks  ...  K1 16 blocks  K1 4 blocks  K2 16 blocks  ...

Kernel slicing

Timeline

K1 6 blocks  K1 6 blocks  K1 6 blocks  K1 6 blocks  ...
K2 10 blocks  K2 10 blocks  K2 10 blocks  K2 10 blocks  ...

* Improving GPGPU energy-efficiency through concurrent kernel execution and DVFS - Jiao, Qing, et al.