Benchmark Suite Construction for Multicore and Accelerator Architectures

Kevin Skadron
University of Virginia

Context

- The multicore era has spawned massive diversity
  - 2,4,6, soon 8-way ILP cores
  - Many simple, multithreaded cores (e.g. Niagara)
  - Heterogeneous organizations
    - GPUs, FPGAs, etc. as accelerators
- How do we design hardware and software?
- Need driving workloads

Session Objectives

- Come up with specific recommendations for future benchmarks
  - Features, research questions
  - Perhaps publish as an article in ACM SIGARCH Computer Architecture News, or even IEEE Computer
- Open discussion
  - Can form brief breakout groups if warranted
- Can have follow-up meetings if sufficient interest

Many Questions! ~~pec~~

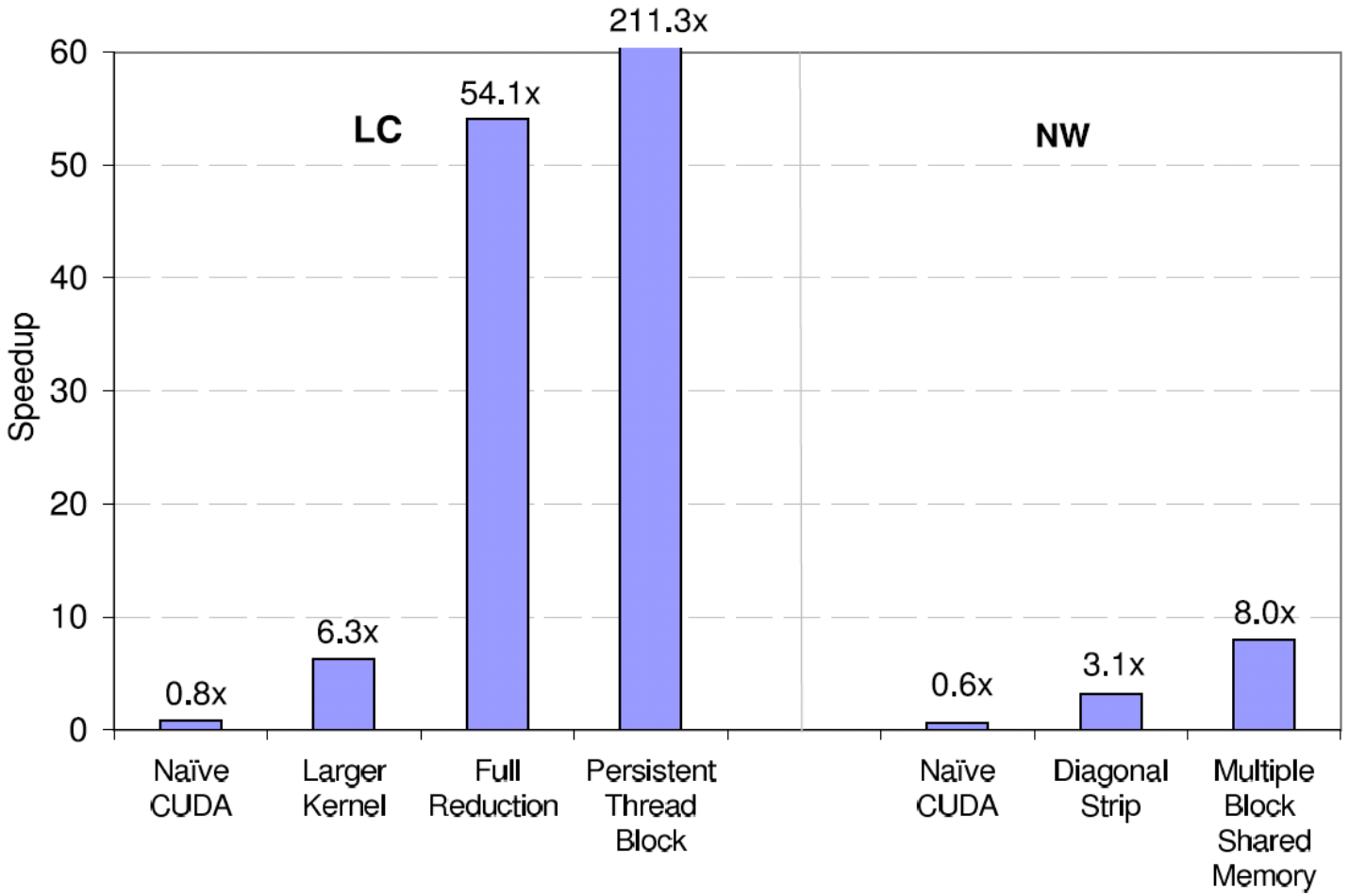
- What is the role of a benchmark suite? Comparison or research?
  - Current or futuristic benchmarks?
- Primarily for HW or SW? Can we use the same benchmark suite?
  - HW: run on native hardware or in simulation?
  - SW: primarily for middleware or end-users?
- How optimized?
  - Trying to capture “average” or “code hero” programming?
  - Maybe both?
- How to make portable?
  - How to deal with current and *future* heterogeneity
  - Porting large applications is expensive
  - If software is optimized for specific hardware details, how to deal with rapid evolution?
- What workloads are “representative”? Are they open source?
- Stressmarks, building blocks, standalone applications, workflows?
- What language(s)?
- Metrics?
- Support for simulation?

Etc….

A Few Examples

- Arch research
  - Identify bottlenecks in current hardware
  - Propose new features or architectures
- Compiler research
  - Code generation
  - Parallelization
- Software development
  - Templates/exemplars

Incremental Performance Improvement



- Which optimization to apply and the order to apply optimizations is not always intuitive
- Some optimizations are unlikely to be discovered by the compiler

	Parsec	SPLASH-2	Rodinia (IISWC'09)
Platform	CPU	CPU	CPU and GPU
Programming Model	Pthreads, OpenMP, TBB	PARMACS macros	OpenMP, CUDA
Machine Model	shared memory	shared memory	shared memory, offloading
Application Domain	scientific/engineering, finance, multimedia	scientific/engineering, graphics	scientific/engineering, data mining
NO. of Applications	3 kernels, 9 apps	4 kernels and 8 apps	6 kernels and 5 apps
Optimized for	multicore	distributed shared memory multiprocessor	manycore, accelerator
Incremental Opt. Ver.			
Memory Space	HW cache	HW cache	HW/SW cache
Problem Sizes	small - large	small - medium	small - large
Special SW techniques	SW pipelining	NA	ghost-zone, persistent thread-block
Synchronization	barrier/lock/condition	barrier/lock/condition	barrier

Brief Characteristics of Other Suites

	Multithreaded	Domain-specific	Model	Platform
SPEC CPU 2006			NA	CPU
SPEC OMP 2001			OpenMP	CPU
ALPBench			Pthreads	CPU
Biobench			NA	CPU
BioParallel			OpenMP	CPU
MediaBench			NA	CPU
MineBench			OpenMP	CPU
Parboil			CUDA	GPU

Questions for the Participants

- What important features are missing?
- What is needed for heterogeneous computing?
- How much should benchmarks focus on the most challenging (vs. most common)?
  - e.g., doall vs. fine-grained, irregular parallelism