CScADS Summer 09 Workshop
Libraries and Autotuning

Panel on Challenges in Industrial Software Development

Kevin Wadleigh
Math Libraries and Applications
Convey Computer Corporation
CScADs – Challenges in industrial software development

• What are the real development and tuning problems in industrial software libraries?
• What are the major challenges in bringing autotuning techniques into industrial software libraries?
First generation of autotuning

• If done well, autotuning allows good performance across architectures

• Original efforts attempted to do common problems – GEMM, FFTs

• GEMM - PhiPAC
  – no updates in a while

• GEMM - ATLAS
  – Still being update/maintained, expanded beyond GEMM
  – Contrast with Goto’s library – also freely available, popular GEMM, but not autotuned

• FFTW
  – Users like it because it is the only high quality FFT package with a consistent API that performs well on multiple platforms
  – It is a poor reflection on our community that there is not a standard FFT API such as LAPACK

• Some users use these packages because they’re free
What should autotuning address?

- Changes in computer companies in 2009
  - Status of SGI, Sun, SiCortex, ClearSpeed

- How many vendors produce math libraries?

- How many standard architectures are left?
  - Autotuning was originally developed to address the many RISC vendors, vector computers, and others 10+ years ago

- Standard microprocessors have become unified
  - x86 derivatives dominate HPC now
  - 87% of June 2009 Top500 list use x86 type processors

- Autotuning should at least address
  - multiple cores (using OpenMP or pthreads)
  - distributed computing (MPI) (assuming there’s enough work to do)
Heterogeneous Systems

• Interesting new architectures are heterogeneous
  – GPUs (Nvidia, AMD, Intel)
  – Hybrid core FPGAs (Convey)
  – ASICs (ClearSpeed)

• Efforts like MAGMA (Matrix Algebra on GPU and Multicore Architectures) are needed to encompass heterogeneous architectures
  – A Note on Auto-tuning GEMM for GPUs (Li, Dongarra, Tomov) indentifies characteristics of algorithms to be discussed
    • High parallelism
    • Reduced communication
    • Heterogeneity-awareness
Power of an Integrated System Architecture

- Programmed in standard C, C++, and Fortran
- Leverages x86/Linux ecosystem
- Coprocessor extends x86 ISA
Characteristics from MAGMA and comments about Convey (1)

• High parallelism (to efficiently account for many-cores)

• Different personalities have different characteristics – example single precision personality
  – 4 FPGAs
    • 8 FMA pipes per FPGA which can each do 4 fmas
    • 8 MISC pipe per FPGA which can each do 2 adds and other ops
    • So 128 fma and an additional 64 adds all simultaneously
  – Sixty-four 64-bit vectors of length 1024
  – 8192 outstanding loads so that portions of many vectors can be in flight at any one time
  – It’s vectorization – use the compiler
SPvector Personality

A load-store vector architecture with modern latency-hiding features

Optimized for Signal Processing (i.e., Oil & Gas) applications

4 Application Engines / 32 Function Pipes
vector elements distributed across function pipes

Same instructions sent to all function pipes
Each function pipe supports:
− multiple functional units
− out-of-order execution
− register renaming
Characteristics from MAGMA and comments about Convey (2)

• *Reduced communication (to account for increasing memory wall)*

• Convey’s architecture is a cache coherent NUMA system with multiple connections
  - Coprocessor has 80 GB/s memory bandwidth
  - Host (Intel®64) has standard memory bandwidth
  - Automatic cache coherency handles communication between the two on standard front side bus
    • Users can specify memory placement using numaclt, first touch policy, function calls, or pragmas
  - Clustering using host interconnect (InfiniBand, 10GigE) and MPI
Convey System Architecture

**Intel x86_64**
*Linux ecosystem*

Commodity Motherboard

- Intel Processor
- Intel chipset
- PCIe
- 4 DIMM channels

Coproprocessor

- Host interface
- Application Engines
- Coherent memory controller
- 16 DIMM channels (80GB/sec)
- ECC memory

**Dynamically loadable personalities deliver application specific performance**

**Shared physical and virtual memory provides coherent programming model**
Characteristics from MAGMA and comments about Convey (3)

- **Heterogeneity-awareness** (*split algorithms between components*)
- In a heterogeneous system how much work should be done by which processor?
  - Dynamic selection: small problems done on x86, large ones on coprocessor; when to use both?
- How does one handle varying ratios of x86 and coprocessors?
- Convey can be reconfigured to many different personalities (think processors), so there will need to be autotuning for each type
Final thoughts

• Vendor software is key
  – Compiler
    • Must have a mature compiler for autotuning
    • One compiler per architecture (as on Convey) or many?
  – Other considerations
    • Convey has virtual memory, a single memory space per computer, cache coherency and is NUMA aware – not true for most accelerators

• Any autotuned industrial libraries should be applicable to multiple disciplines, such as
  – GEMM and FFTs from first generation
  – Direct dense, direct sparse, iterative sparse solvers

• Higher levels of abstraction are needed to increase amount of work
  – This conference has been focused on massively parallel systems, yet most autotuning efforts use at most a few cores

• Convey’s reconfigurability exposes new opportunities for autotuning