Center for Scalable Application Development Software

CScADS Workshop on Performance Tools for Petascale Systems

John Mellor-Crummey and Bart Miller
Department of Computer Science
Rice University

DOE SciDAC Program

- Portfolio of coordinated research efforts directed at exploiting the emerging capabilities of terascale and petascale computing
- These research projects respond to
  - the extraordinary difficulties of realizing sustained peak performance for scientific applications that require terascale and petascale capabilities to accomplish their research goals
  - the need for developing collaborative software environments where distributed resources and expertise are combined to address complex questions that no single institution can manage alone

http://www.scidac.org

DOE SciDAC-2 Mission

- Develop comprehensive scientific computing software infrastructure to enable petascale science
- Develop new generation of data management and knowledge discovery tools for large data sets

DOE SciDAC-2 Program Investments

- Enabling technologies
  - computer science
  - applied math
  - visualization and data mgmt.

- Science application areas
  - physics
  - climate
  - groundwater
  - fusion energy
  - life sciences
  - materials and chemistry

Participants
- 17 labs
- 55 universities
- 3 companies

Partners

Rice University (Kennedy (late), Mellor-Crummey, Cooper)

Argonne National Laboratory (Beckman, Lusk, Gropp)

University of California - Berkeley (Yelick)

University of Wisconsin - Madison (Miller)

University of Tennessee (Dongarra)

Scope of Activities

- Community outreach and vision building
- Research and development
- Open source software infrastructure
  - *compiler infrastructure*
    - support high-level source-to-source optimization of programs
      - leverage Rose, Open64, Rice's D System compiler infrastructure
  - *performance tools infrastructure*
    - support binary analysis, instrumentation, data collection, and measurement interpretation
      - leverage Rice's HPCToolkit and Wisconsin's Paradyn & Dyninst tools

CScADS Research Themes

- Rapid construction of high-performance applications
  - compiler technology for scripting languages and component composition
- Scaling to homogeneous parallel systems
  - tools for parallel performance analysis and improvement
  - compiler technology for parallel languages
    - partitioned global address space (PGAS) languages
    - global array languages
    - parallel scripting languages
  - support for multicore platforms
    - decomposing and mapping parallelism to available resources
    - transforming applications to reuse data wherever possible
    - choreographing parallelism and data movement
- Portability and support for heterogeneous platforms
  - automatic tuning to new platforms
  - compiling to heterogeneous platforms

Workshop Technical Focus

Workshop Charge

- Identify important open problems and challenges for performance tools for petascale systems
- Brainstorm on promising approaches to open problems
- Identify infrastructure needs to address key challenges
- Assess available infrastructure
- Identify opportunities for synergy
  - opportunities to
    - consolidate and harden existing infrastructures
    - reuse existing components developed by others
    - refactor and extend existing components to apply them to new challenges
- Collaborate on design of sharable components

Workshop Structure

- 1.5 days of presentations
- Tomorrow afternoon
  - general discussion
    - identify candidates for working groups
    - interests and group membership
  - informal presentations of “hot button” issues?
- Late Tuesday, Wednesday morning
  - working groups meet
- Wednesday afternoon
  - informal discussion of research challenges, ongoing tools research, working group activities, and collaboration plans
- Thursday
  - working groups meet
  - summary of working group activities and open discussion

Performance Tools for Petascale Systems

Goal: provide insight into how to improve programs for better performance on petascale systems that are ensembles of multicore microprocessors

Classes of issues

- Understanding executables
- Instrumentation
- Measurement
- Analysis
- Modeling
- Presentation

Understanding Executables

- Support for machine-code based
  - instrumentation
  - measurement
  - analysis
- Understand instruction stream
  - mix
  - dependencies
  - delays
  - memory accesses
  - instantaneous state at arbitrary points in the execution
    - e.g. frame state for asynchronous unwinding
- Support work with stripped code, e.g. 3rd party libraries

Instrumentation

- Goals
  - correctness
  - efficiency
  - portability
- Instrumentation approaches
  - binary rewriting
  - binary patching
  - dynamic compilation
- Open problems?

Measurement Challenges

- Multi-scale parallelism
  - ILP, SIMD units, multi-core processor, intra-node, inter-node
- Understanding processor core activity
  - utilization of functional units
  - utilization of memory hierarchy
    - memory parallelism
    - data reuse or lack thereof: temporal, spatial
  - attributing stalls to causes
- Understanding parallelism
  - overheads
    - communication: latency, bandwidth
    - serialization
    - load imbalance
    - contention
      - cache blocks, network links

Measurement Techniques

- Approaches
  - tracing
  - sampling
    - flavors: node-based vs. communication-based
    - calling context or not
  - traces of samples

- Issues
  - must capture meta-data
  - completeness
  - fidelity
  - efficiency
  - data volume
  - utility

Analysis Challenges 1

- Diagnosing processor utilization inefficiencies
  - instruction mix underutilizes core: improve instruction balance
  - slack instruction schedule
  - memory subsystem inefficiencies
    - too many cache, TLB misses
    - why are we not getting better bandwidth?
      - inadequate memory parallelism?
      - not enough reuse of memory pages?
    - opportunities for reducing bandwidth by improving data reuse?
    - opportunities for hiding latency?
- Diagnosing multithreading deficiencies
  - are the cores/processors underutilized; if so, why?
      - less runnable threads than processors
        - fewer threads than processors or blocked threads?
  - is the system thrashing (time slicing too many active threads)

Analysis Challenges 2

- Diagnosing parallelism deficiencies
  - load imbalance
  - serialization
  - communication overhead
    - communication granularity/frequency
    - exposed data transfer costs
  - strategies for data mining
    - clustering
    - anomaly detection
- Understanding the performance of hybrid parallelizations

Modeling

Understand potential for improving performance by using alternate

- hardware architectures
- data structures
- computation structures

Presentation

- Textual-based approaches
- Scalable visualizations