CScADS Workshop on Performance Tools for Petascale Systems
Analysis and Presentation
Working Group Notes
Working Group Members

- Jim Brandt
- Brad Chen
- Luiz DeRose
- Bronis de Supinski
- Stephane Eranian
- Robert Fowler
- James Galarowicz
- Marty Itzkowitz
- Karen Karavanic
- Jesus Labarta

- Allen Malony
- John McCalpin
- Philip Roth
- Martin Schulz
- Seetharami Seelam
- Sameer Shende
- Dan Terpstra
- Jeffrey Vetter
- Felix Wolf
What to display?

- Driven by analysis goals?
- Driven by available data?
- Driven by user category?
  - domain scientist vs. analyst
Single Node Performance

- Capture metrics associated with code locations
  - source
  - executable
- Attribution to context
- Ex compiler output valuable – information about what transformations have occurred
  - attributing costs back to source level constructs
  - correct instruction (e.g. vs. following instruction)
  - how to meaningfully present the result?
  - still useful if tool user isn’t familiar with the code?
- Counters – including measurements in user space
Application Abstractions

- Models
- Algorithms
- Expected behavior
Data Format Alternatives

- XML
- Raw data for max flexibility and model development
Barriers to Tool Adoption 1

• **Ease of Use**
  – so many options, how to choose best display for the task?
  – tool as swiss army knife with 18 specialized blades

• **Users may not trust the tool answer**
  – users unfamiliar with underlying measurement techniques/limits
  – users unfamiliar with statistical methods, how to interpret output?

• **Users need to know the answer to choose the best approach**
  – better for tool to pick or at least focus choices or suggest
  – user’s initial guess may be wrong

• **Tools don’t include knowledge of application the developer has – algorithm etc**

• **Perspective: analysis probably won’t improve performance**
  – performance “good enough”
  – problem is the underlying machine – app is doing best possible
Barriers to Tool Adoption 2

• Hybrid machine -> load imbalance
• Machine “upgrade” -> drop in performance
  – performance vs scaling
• Users can’t do cost/benefit analysis up front
  – cost of analysis?
  – potential gain?
  – will improving performance benefit app developer/user?
  – analysis may be more valuable to system purchasers, designers of next generation machines, operators, researchers
  – better to get high level answer first to decide to analyze more closely
• Provide automatic “high level report card”
  – one approach: don’t charge runs to user if monitoring tool used
  – ignorance
    • what tools are available?
    • what might they show?
Barriers to Tool Adoption 3

• Security / access issues
  – “need to know”
  – app developers/analysts may hold different levels of clearance
  – may rule out continuous system monitoring or mandatory measurement
  – tool developers need full scale measurement data

• Production scale runs are expensive
  – management must be convinced of the potential benefit of performance tuning applications (2% overhead to monitor vs ??)
    – only way to find some problems – small scale tests miss things
Methods for Analysis

- Anomaly detection
- Dimension reduction
- Comparative Analysis (cf to previous runs; models; expected behavior)
  - requires a data store for historical knowledge
- Multiple experiments --> a single execution view
- Clustering
- Principal component analysis
- Pattern identification
  - difficulties for pattern analysis
    - patterns can arise from superposition of phenomena
    - determining the reason for patterns is hard
  - especially with respect to temporal data
  - phase identification to reduce data size
  - multi-scale pattern analysis
    - bottom up and top down
    - potential role for simulation?
    - what if analysis (if pattern were different)
Strategies for Monitoring and Analysis

- Continuous System level monitoring (eg NWPerf @PNNL)
- Parallel data analysis methods – e.g. SCALASCA
Methods for Scalable Display 1

- **Data Reduction**
  - can’t show all the data anymore – even at terascale
  - tracefile size constraints
  - analysis must occur before display
  - anomaly detection

- **Large multi-dimensional parameter space**

- **Integrating historical data into display**

- **Thumbs up/thumbs down**
  - very short answer from automated approach

- **Common interfaces**
  - component reuse between tools
  - Eclipse?
    - “Eclipse aware” more flexible than requiring Eclipse
  - well defined formats for input/output data
Methods for Scalable Display 2

- Maximizing portability
  - use of file system: ability to output tool results to files
  - Java, QT, SWT, ...
  - support for multiple DBMSs

- New techniques to take advantage of large performance database archives?

- Minimizing data movement
  - client/server approach
  - just in time data delivery

- Using data visualization tools developed for app data
  - analogy to car performance display
    - click on component, get detailed info