Analyzing the Performance of Parallel Applications (on IBM Systems) with KOJAK

Bernd Mohr

Forschungszentrum Jülich (FZJ)
John von Neumann Institute for Computing (NIC)
Central Institute for Applied Mathematics (ZAM)
52425 Jülich, Germany

b.mohr@fz-juelich.de
Basic Idea

- "Traditional" Tool
  - For non-standard / tricky cases (10%)
  - For expert users

- Automatic Tool
  - For standard cases (90% ?!)
  - For "normal" users
  - Starting point for experts

⇒ More productivity for performance analysis process!
The KOJAK Project

- Kit for **Objective** Judgement and **Automatic** Knowledge-based detection of bottlenecks
- Forschungszentrum Jülich
- Innovative Computing Laboratory, TN
- **Long-term goals**
  - Design and Implementation of a Portable, **Generic**, and **Automatic** Performance Analysis Environment

**Approach**

- **Instrument** C, C++, and Fortran parallel applications
  - Based on MPI, OpenMP, SHMEM, or hybrid
- **Collect** event traces
- **Search** trace for event patterns representing inefficiencies
- **Categorize and rank** inefficiencies found

KOJAK: Supported Platforms

- Instrumentation and measurement only (analysis on front-end or workstation)
  - IBM BlueGene/L
  - Cray T3E, Cray XD1, Cray X1, and Cray XT3
  - Hitachi SR-8000
  - NEC SX
  - Fujitsu Primepower (Sparc, Solaris)

- Full support (instrumentation, measurement, and automatic analysis)
  - Linux IA32, IA64, and EMT64/x86_64 based clusters
  - IBM AIX Power3/4/5 based clusters (SP2, Regatta)
  - SGI Irix MIPS based clusters (Origin 2K, Origin 3K)
  - SGI Linux IA64 based clusters (Altix)
  - SUN Solaris Sparc and x86/x86_64 based clusters (SunFire, …)
  - DEC/HP Tru64 Alpha based clusters (Alphaserver, …)
Example Performance Property: Wrong Order Late Sender

Location vs. Time diagram showing:
- SEND operations marked at different locations (B, C)
- RECV operations showing waiting time
- Time axis indicating sequential order
KOJAK Instrumentation and Measurement

1. Instrument user application with EPILOG tracing library calls
   - User functions and regions:
     - Preferred: Manually using POMP directives
     - Also: Manually using EPILOG tracing API functions
     - Also: Automatically by Compiler
   - MPI calls:
     - Automatically by PMPI Wrapper Library (MPI 1.2 + MPI 2 RMA)
   - OpenMP
     - Automatically by OPARI source code instrumentor
   - Alternative: Configure TAU to generate EPILOG traces

2. Running instrumented executable will produce trace file named prefix.elg
   - Environment variables allow to control the measurement
KOJAK Automatic Analysis Process

3 expert <elg-trace-file>
   - Scans event trace for patterns; if instance found:
     - Determine call path and process/thread affected
     - Calculate severity ::= percentage of total execution time “lost” because of pattern
   - Result stored in “cube” file containing
     - For each pattern: distribution of severity
       - Over all call paths
       - Over machine / nodes / processes / threads

4 cube <cube-file>
   - Result presentation via three interconnected tree browsers
     - Pattern hierarchy (general ⇒ specific problem)
     - Region / call tree
     - HW + SW location hierarchy
Other Features and Future Work

- Other major features
  - Comparative analysis of multiple executions
    - Merge, Mean, and Difference of experiments
  - Holistic analysis
    - Combined hardware counter metrics + time-based analysis
  - Extensive one-sided communication analysis
    - MPI-2 RMA and SHMEM (CAF and UPC soon?)
  - Interoperability
    - Work with TAU, Vampir, and Paraver groups

- Current Research: SCALABILITY
  - Scalable trace format
  - Parallel pattern analyzer
  - Scalable displays
CUBE Performance Algebra

- "Compute" with CUBE result files
  - Mean
  - Merge
  - Diff

- Raised relief / positive values show improvement
- Sunken relief / negative values show degradation
Holistic Performance Analysis

- Collect
  - series of experiments
  - with different but matching sets of hardware counters

- Merge results using
  - Generic
  - Platform-specific counter hierarchy specification
**KOJAK ↔ TAU ↔ VAMPIR ↔ Paraver**

- **TAU** (University of Oregon)
  - Comprehensive performance analysis framework and toolset
  - Can produce profiles and event traces
  - Supports a large variety of languages / paradigms / platforms
  - [http://www.cs.uoregon.edu/research/tau/](http://www.cs.uoregon.edu/research/tau/)

- **Vampir** (Technical University of Dresden)
  - Well known and easy-to-use event trace visualization tool
  - [http://www.vampir.eu](http://www.vampir.eu)

- **Paraver** (Barcelona Supercomputing Center / CEPBA)
  - "Programmable" event trace visualization tool
    - Complex analysis and filter functions
    - Supports definition of derived metrics
  - [http://www.cepba.upc.es/paraver/](http://www.cepba.upc.es/paraver/)
KOJAK

http://www.fz-juelich.de/zam/kojak/
kojak@fz-juelich.de
Factors leading to large traces

- **Temporal coverage**
  - full
    - disabled
  - partial

- **Granularity / event rate**
  - high
  - low

- **Problem size**
  - large
  - small

- **Number of event parameters**
  - many
  - few

- **Number of processes**
  - width
Sequential Analysis Process

Instrumentation

Source Code → Automatic Multilevel Instrumentation → Executable

Measurement

Execution on Parallel Machine → Local Definition & Trace Files

Analysis

Merge → Global Definition & Trace File → Sequential Analyzer (EXPERT) → Global CUBE File
Scalable Automatic Trace Analysis

- Parallel pattern search to address wide traces
  - As many processes / threads as used to run the application
    ⇒ Can run in same batch job!!
  - Each process / thread responsible for its “own” local trace data

- Idea: “parallel replay” of application
  - Analysis uses communication mechanism that is being analyzed
  - Use MPI P2P operation to analyze MPI P2P communication,
    use MPI collective operation to analyze MPI collectives, ...
  - Communication requirements not significantly higher and (often lower) than requirements of target application

- In-memory trace analysis
  - Available memory scales with number of processors used
  - Local memory usually large enough to hold local trace
Parallel Analysis Process

Instrumentation

Source code → Automatic Multilevel Instrumentation → Executable

Measurement

Automatic Multilevel Instrumentation → Execution on Parallel Machine

Execution on Parallel Machine → Local Definition & Trace Files

Analysis

Local Definition & Trace Files → Parallel Analyzer

Parallel Analyzer → Global CUBE File
Current Prototype (July 2006)

Instrumentation

Source code → Automatic Multilevel Instrumentation → Executable

Measurement

Local Trace Files → Execution on Parallel Machine → Local Definition Files

Analysis

ID Mapping Tables → Unification

Parallel Analyzer → Local Results → Combine

Global CUBE File
Example Pattern: Late Sender

- **Sequential approach (EXPERT)**
  - Scan the trace in sequential order
  - Watch out for receive event
  - Use links to access other constituents
  - Calculate waiting time

- **New parallel approach (SCOUT)**
  - Each process identifies local constituents
  - Sender sends local constituents to receiver
  - Receiver calculates waiting time
Example: sweep3D on 8192 BG/L PEs

- New topology display
- Shows distribution of pattern over HW topology
- Scales to larger systems
Example: smg2000 on 16384 BG/L PEs
Some Numbers for the 16K CPU Execution

- SMG2000, 64x64x32 grid size, 3 iterations
- Trace files size
  - 0.5 TByte (0.25 TByte compressed)
  - 40 x 10e9 events
- Execution times
  - SMG2000: 40s
  - Dumping 16K traces: 50min
  - Reading 16K traces: 14min
  - Parallel analysis: 2 min 10s
  - Writing 16K local results: 7 min 10s
  - Combine local results: 2085 CPU/h (~16h on four 32way nodes p690)

⇒ Current prototype good for up to 2048 (4096) CPU runs
⇒ Execution times with a couple of minutes
SCALASCA

http://www.scalasca.org/
Current (almost complete) KOJAK Team

Brian Wylie FZJ
Matthias Jurenz TUD
Bernd Mofir FZJ
Markus Geimer FZJ
Daniel Becker FZJ

Christoph Geile FZJ
Holger Brunst TUD
Farzona Pulatova UT
Marc-Andre Hermanns FZJ
Felix Wolf FZJ