UPC Performance Analysis Tool: Status and Plans

Professor Alan D. George, Principal Investigator
Mr. Hung-Hsun Su, Sr. Research Assistant
Mr. Adam Leko, Sr. Research Assistant
Mr. Bryan Golden, Research Assistant
Mr. Hans Sherburne, Research Assistant

HCS Research Laboratory
University of Florida
Outline

- Accomplishments
- PAT High-level Design
- PAT Development Plan
- Q&A
Accomplishments

- Survey of PAT technology
- Evaluation of existing parallel PAT using a standardized scoring system
- Study of UPC language with respect to performance analysis
- Usability study
- Identification of reusable components
- Requirements for PAT
- High-level design for PAT
Performance Analysis Technologies

- Three general performance analysis approaches
  - Analytical modeling
    - Mostly predictive methods
    - Could also be used in conjunction with experimental performance measurement
    - Pros: easy to use, fast, can be done without running the program
    - Cons: usually not very accurate
  - Simulation
    - Pros: allow performance estimation of program with various system architectures
    - Cons: slow, not generally applicable for regular UPC/SHMEM users
  - Experimental performance measurement
    - Strategy used by most modern PATs
    - Uses actual event measurement to perform the analysis
    - Pros: most accurate
    - Cons: can be time-consuming

PAT = Performance Analysis Tool
Tools Study: List of Tools

- Profiling tools
  - TAU (Univ. of Oregon)
  - mpiP (ORNL, LLNL)
  - HPCToolkit (Rice Univ.)
  - SvPablo (Univ. of Illinois, Urbana-Champaign)
  - DynaProf (Univ. of Tennessee, Knoxville)

- Tracing tools
  - Intel Cluster Tools (Intel)
  - MPE/Jumpshot (ANL)
  - Dimemas & Paraver (European Ctr. for Parallelism of Barcelona)
  - MPICL/ParaGraph (Univ. of Illinois, Univ. of Tennessee, ORNL)

- Other tools
  - KOJAK (Forschungszentrum Jülich, ICL @ UTK)
  - Paradyn (Univ. of Wisconsin, Madison)

- Also quickly reviewed
  - CrayPat/Apprentice2 (Cray)
  - DynTG (LLNL)
  - AIMS (NASA)
  - Eclipse Parallel Tools Platform (LANL)
  - Open/Speedshop (SGI)
Tools Study: Lessons Learned

- Most effective tools present data to user at many levels
  - Profile and summary data that gives top-level view of program’s execution
  - Trace data that shows exact behavior and communication patterns
  - Hardware counters that show how well program runs on current architecture

- Common problems experienced
  - Difficult installations
  - Poor documentation, no “quickstart” guide
  - Too much user overhead
  - Not enough data available from tools to troubleshoot performance problems
  - Difficult to navigate tool to get to performance data or nonstandard user interfaces
  - Inability to relate data back to source code
Overall Scores

HPCToolkit 50.375
MPICL 38.25
SvPablo 48.625
ICT 55.875
KOJAK 60.625
Dynaprof 42.5
mpiP 47
DiP 42.625
MPE/Jumpshot 48
Paradyn 50.75
TAU 62
Language Study & Usability Study

Language study
- Hybrid pre-processor/wrapper library approach appears appropriate
  - UPC functions → wrappers
  - Others (direct memory access, upc_forall, etc.) → pre-processor
- UPC profiling interface
  - Avoid prevention of compiler optimization
  - Obtain low-level / implementation specific data

Usability study
- Elicit user feedback through a Performance Tool Usability Survey – limited success thus far
  - [http://www.hcs.ufl.edu/prj/upcgroup/survey.html](http://www.hcs.ufl.edu/prj/upcgroup/survey.html)
- Review and develop a concise summary of literature regarding usability for parallel performance tools
Extensibility & Code Reuse

- Instrumentation & measurement units
  - Source instrumentation: tau_instrument, PDToolkit
  - Binary instrumentation: DynInst, DPCL, ATOM, Pin
  - Hardware counters: PAPI, PCL
  - General trace and profile routines: TAU, EPILOG & EARL from KOJAK
  - Use PAPI, develop own instrumentation technique (UPC profiling interface)

- Analysis unit
  - Source code correlation from wrapper libraries: HPCToolkit, mpiP
  - Pattern matching for bottleneck identification: EXPERT from KOJAK
  - Borrow idea + develop one-sided communication model

- Presentation unit
  - Trace visualization: Jumpshot, Intel Trace Analyzer or VampirNG using libvtf
  - Profile visualization: CUBE from KOJAK, Paraprof from TAU
  - Source code viewer: ToolGear, HPCToolkit
  - Develop on our own but borrow small pieces of code

- UPC PAT → Develop new tool that borrows ideas from existing tools
PAT High-level Design

- Semi-automatic source-level instrumentation as default
  - Minimize user workload
  - Reduce development complexity
  - Provide easy source-code correlation

- Tracing mode as default with profiling support

- Analyses: load balancing, scalability, memory system

- Visualizations:
  - Timeline display
  - Speedup chart
  - Call-tree graph
  - Communication volume graph
  - Memory access graph
  - Profiling table
PAT High-level Design: Experimental Performance Measurement Stages

- **Instrumentation** – user-assisted or automatic insertion of instrumentation code
- **Measurement** – actual measuring stage
- **Analysis** – filtering, aggregation, analysis of data gathered
- **Presentation** – display of analyzed data to user, deals directly with user
- **Optimization** – process of finding and resolving bottlenecks

Original code → Instrumentation → Measurement → Analysis → Presentation → Optimization → Analytical modeling → Simulation
PAT High-Level Design – Simplified Version

M UNIT

Measurement modules

Automatic source

Low-level event read module

UPC Prof
# Research and Release Schedule

- Beta-tester needed!!

<table>
<thead>
<tr>
<th>PAT Release</th>
<th>Delivery Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>I and M modules for 1st platforms</td>
<td>April 1, 2006 (Month 6)</td>
</tr>
<tr>
<td>I, M, and A modules for 1st platforms</td>
<td>July 1, 2006 (Month 9)</td>
</tr>
<tr>
<td>I and M modules for 2nd platforms</td>
<td>July 1, 2006 (Month 9)</td>
</tr>
<tr>
<td>Beta-1 (I, M, A, and P modules for all platforms)</td>
<td>October 1, 2006 (Month 12)</td>
</tr>
<tr>
<td>Beta-2</td>
<td>December 1, 2006 (Month 14)</td>
</tr>
<tr>
<td>Beta-3</td>
<td>February 1, 2007 (Month 16)</td>
</tr>
<tr>
<td>Version 1.0</td>
<td>April 1, 2007 (Month 18)</td>
</tr>
</tbody>
</table>