Metadata Collection for Performance Analysis

Karen L. Karavanic
Associate Professor of Computer Science
Portland State University
The PerfTrack Project

- PerfTrack is a tool for storing, exploring, and analyzing performance data
- Our Approach:
  - Collect and store as much information as possible about each build and run of an application
  - Integrate database technology into a performance analysis tool
  - Store a wide variety of performance data
    - Data from different measurement tools
      - Tracing, DPCL, Paradyn, TAU, Vampir, Speedshop, HW counters, etc.
    - Native application performance measurements
PerfTrack Design

- Design goal: No knowledge of database technology or vocabulary should be required to use PerfTrack
- Design goal: PerfTrack must be scalable to 1000s of program runs and 100,000s of performance results
- Design goal: PerfTrack must be flexible enough to store data from different measurement tools and different types of performance studies
- Design goal: PerfTrack must be extensible to accommodate future innovations in measurement and analysis
- Design goal: PerfTrack should not be limited to a specific DBMS package for its data store
PerfTrack Design

- **Data Collection Scripts:**
  - Build environment
  - Run environment
  - Performance Data

- **PTdataStore interface:**
  - Shelters PerfTrack user from the DBMS
  - Perftrack Data Format (PTDF)

- **Data navigation and analysis**
  - PerfTrack GUIs
  - Command line interface
  - Direct SQL query
PerfTrack Design

• Performance Result: a measured value
  - Metric: cpu time, wall clock time, wait time
  - Context: whole program? One function? One process?
  - Value: result of measuring a metric in a context
• Everything is a resource: machine, node, process, function, execution ...
  - Resource attribute:
    • Execution - timestamp, build, machine, etc.
    • Compiler - version
  - Resource type
    • Hierarchical or non-hierarchical
Build Application

ptbuild.py --app umt2k --pathToExe ./umt2k --srcDir . -V

This collects information about the build, such as the machine the application was built on, the compilers used, and environment variables that were set during the build.
Build Attributes

- **BUILD**: Date, Time, Environment Variables
- **BUILD/MODULE**: Path, Size, Timestamp
- **COMPILER**:
- **ENVIRONMENT/MODULE**: LibraryDynamic, Path, Size, Timestamp, Type
Execute Application

ptrun.py --app umt2k  --batchFile psub.script  --inputDeck “opacfile,rtin,smartin”
--exeName ./umt2k

This collects information about the execution, such as the machine the application was run on, and environment variables that were set during the execution.

environment
- module
  - Path to library
  - Size
  - Timestamp
  - Type

operatingSystem
- Name
- Release
- Version

machine
- Name

node
- Name

execution
- App-specific values
- Permissions on exe
- Time stamp
- Number of processes
- Number of threads
- Concurrency model (MPI, OpenMP)
- Environment variables
- Languages in application
- User who ran application

inputDeck
- Name
- Mod time

submission
- Time stamp
- Name of batch file
- Commands in batch file
- Run command
- Environment variables set in batch file

processes
- Rank

threads
- Rank

"Metadata Collection for Performance Analysis"
Karen L. Karavanic July 17, 2007
Execution Attributes

- Concurrency
- Environment Variables
- Executable:
  - GID, Name, Permissions, Size, Timestamp, UID, j
- Job:
  - CompletionTime, Exit Status, Nodes, Resources Used, StartTime
- Languages
- Launch Date, Time
- NumberOfProcesses
- PageSize
- ProcessesPerNode, ThreadsPerProcess
- RunErrorMsg # any error messages from the job
- Username
- UsesMPI/OpenMP/Pthreads
## PerfTrack Design: Generic Database Schema

### resource_item
- **id**: Integer
- **name**: Varchar2(255)
- **type**: Varchar2(255)
- **type_id**: Integer
- **parent**: Integer

### resource_attribute
- **res_id**: Integer
- **name**: Varchar2(255)
- **value**: Varchar2(255)
- **type**: Varchar2(255)

### resource_constraint
- **from**: Integer
- **to**: Integer
PerfTrack Design

• PerfTrack Data Format (PTdf):

  ResourceType resourceTypeName
  Application appName
  Execution execName appName
  Resource resourceName resourceTypeName execName
  Resource resourceName resourceTypeName
  ResourceAttribute resourceName attributeName
  attributeValue attributeType
  ResourceConstraint resourceName1 resourceName2
  PerfResult execName resourceSet perfToolName
  metricName value units startTime endTime
Flexible Schema: PERI project example

Base Resource Types
• Grid
• Build
• Environment
• Compiler
• Time
• Operating System
• Execution
• Input Deck
• Preprocessor
• Performance Tool
• Submission
• Metric

Custom resource types
• File System
• FileSystem/device

Custom Attributes
• Submission: Batch queue Entry
• Submission: PBS resources
Submission Attributes

- batchCmd  # commands in the batch file
- batchFile
  - batchFileDateTime
- batchQueueEntry  # entries in the queue at the time of submit
- launcher
  - launcherVersion
- machinePartition
- PBS Resources
- runCmd  # run commands in the batch file
Choose resource names and attributes to search for in the left panel. Add them to the Selection Parameters, then press Get Data to retrieve results.
### Resource Information

**Attributes for resource: IP7.13_rmd-158**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Env_USER</td>
<td>kmohr</td>
</tr>
<tr>
<td>Env_VENDOR</td>
<td>use</td>
</tr>
<tr>
<td>Env_XAUTHLOCALHOSTNAME</td>
<td>jaco01</td>
</tr>
<tr>
<td>Env_XFILESEARCHPATH</td>
<td>/usr/11/L%T%N%C/usr/...</td>
</tr>
<tr>
<td>Execute GID</td>
<td>41710</td>
</tr>
<tr>
<td>Execute Name</td>
<td>/u5/kmohr/minchps_mnp_dyn/13_r...</td>
</tr>
<tr>
<td>Execute Permissions</td>
<td>0755</td>
</tr>
<tr>
<td>Execute Size</td>
<td>1249653</td>
</tr>
<tr>
<td>Execute Timestamp</td>
<td>2007-03-10T08:24:06</td>
</tr>
<tr>
<td>Execute UID</td>
<td>41710</td>
</tr>
<tr>
<td>jobCompletion time</td>
<td>Sun Mar 11 14:39:53 PDT 2007</td>
</tr>
<tr>
<td>jobExitStatus</td>
<td>1</td>
</tr>
<tr>
<td>jobNodes</td>
<td>jaco09</td>
</tr>
<tr>
<td>jobResourcesUsed</td>
<td>supercent=99,cpu=00:01:44,mem=1557,</td>
</tr>
<tr>
<td>jobStartTime</td>
<td>Sun Mar 11 14:38:04 PDT 2007</td>
</tr>
<tr>
<td>Languages</td>
<td>C</td>
</tr>
<tr>
<td>LaunchDateTime</td>
<td>2007-03-10T09:18:04</td>
</tr>
<tr>
<td>NumberOfProcesses</td>
<td>8</td>
</tr>
<tr>
<td>PageSize</td>
<td>4096</td>
</tr>
<tr>
<td>ProcessesPerNode</td>
<td>1</td>
</tr>
<tr>
<td>RunErrorMsg</td>
<td>/usr/common/homes/kmohr/minchps/mnt/...</td>
</tr>
<tr>
<td>RunErrorMsg_2</td>
<td>compiler: Warning: tasks 0-7 exited...</td>
</tr>
<tr>
<td>ThreadsPerProcess</td>
<td>1</td>
</tr>
<tr>
<td>Username</td>
<td>kmohr</td>
</tr>
<tr>
<td>UsesMFI</td>
<td>True</td>
</tr>
</tbody>
</table>

### Execution Resources

<table>
<thead>
<tr>
<th>Resource</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>build</td>
<td>build-153</td>
</tr>
<tr>
<td>build/module</td>
<td>build-153/libcomplex.a</td>
</tr>
<tr>
<td>build/module</td>
<td>build-153/libflap.a</td>
</tr>
<tr>
<td>build/module</td>
<td>build-153/libqqp_common.a</td>
</tr>
<tr>
<td>build/module</td>
<td>build-153/libqqp_d3.a</td>
</tr>
<tr>
<td>build/module</td>
<td>build-153/libqqp_d.a</td>
</tr>
<tr>
<td>build/module</td>
<td>build-153/libqqp_f.a</td>
</tr>
<tr>
<td>build/module</td>
<td>build-153/libqqp_i.a</td>
</tr>
<tr>
<td>build/module</td>
<td>build-153/libqqp_l.a</td>
</tr>
<tr>
<td>build/module</td>
<td>build-153/libqqp_o.a</td>
</tr>
</tbody>
</table>
Select Data

Choose resource names and attributes to search for in the left panel. Add them to the Selection Parameters, then press Get Data to retrieve results.

Resources

- execution

Show resource information

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTsw_rmd-159</td>
<td>execution</td>
</tr>
<tr>
<td>PTsw_rmd-170</td>
<td>execution</td>
</tr>
<tr>
<td>PTsw_rmd-182</td>
<td>execution</td>
</tr>
<tr>
<td>PTsw_rmd-194</td>
<td>execution</td>
</tr>
<tr>
<td>PTsw_rmd-206</td>
<td>execution</td>
</tr>
<tr>
<td>PTsw_rmd-218</td>
<td>execution</td>
</tr>
<tr>
<td>PTsw_rmd-230</td>
<td>execution</td>
</tr>
<tr>
<td>PTsw_rmd-242</td>
<td>execution</td>
</tr>
</tbody>
</table>

Attribute

- Concurrency
- Env_
- Env_ALCLocal_
- Env_CC
- Env_COLORTERM
- Env_COMPILER
- Env_COMPILER...
- Env_CSHEDIT
- Env_CSHREAD
- ...

Add to Selection Parameters

Add Resource Type

Selection Parameters

<table>
<thead>
<tr>
<th>Relatives</th>
<th>Type</th>
<th>Value</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTsw_rmd</td>
<td>execution</td>
<td>PTsw_rmd...6</td>
<td>6</td>
</tr>
</tbody>
</table>

Items matching all parameters: 6

Clear Highlighted Parameters

Clear All Entries

Combine Data

Cancel

Get Data
## Metadata Collection for Performance Analysis

**Karen L. Karavanic**  
July 17, 2007

### Selection Parameters

<table>
<thead>
<tr>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>execution/PT.x03_rmd-158</td>
<td></td>
</tr>
</tbody>
</table>

### Table

<table>
<thead>
<tr>
<th>saved</th>
<th>value</th>
<th>units</th>
<th>metric</th>
<th>label</th>
<th>combined</th>
<th>start_time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td></td>
<td>Average eg/fers for</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>48.04073</td>
<td>seconds</td>
<td>/Time</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>40</td>
<td></td>
<td>/Total_iters</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>176</td>
<td></td>
<td>Average eg/fers for</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>55.01562</td>
<td>seconds</td>
<td>/Time</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1057</td>
<td></td>
<td>/Total_iters</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Add data columns to table by highlighting resources and attributes, then pressing Add... buttons to select them. Press Get Data to retrieve results.

**Available Display Columns**

- Resources
  - File System
  - Operating System
  - Time/Interval

- Attributes

**Selected Display Columns**

<table>
<thead>
<tr>
<th>Retrieved</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>time/interval</td>
</tr>
</tbody>
</table>

**Selected Resources**

**Selected Attributes**

<table>
<thead>
<tr>
<th>Retrieved</th>
<th>Attribute</th>
</tr>
</thead>
</table>

Add Highlighted Resources

Add Highlighted Attributes

Remove Highlighted Columns

Get Data
Metadata Collection for Performance Analysis

Karen L. Karavanic
July 17, 2007

```
Selection Parameters

<table>
<thead>
<tr>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>execution/PT су3.cpu-158</td>
<td></td>
</tr>
</tbody>
</table>
```

<table>
<thead>
<tr>
<th>saved</th>
<th>value</th>
<th>units</th>
<th>metric</th>
<th>label</th>
<th>combined</th>
<th>start</th>
<th>end</th>
<th>time/interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>48.84673</td>
<td>seconds</td>
<td>/Time</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 1</td>
</tr>
<tr>
<td>2</td>
<td>55.01562</td>
<td>seconds</td>
<td>/Time</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 2</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td></td>
<td>average cg_iters for s</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 1</td>
</tr>
<tr>
<td>4</td>
<td>175</td>
<td></td>
<td>average cg_iters for s</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 2</td>
</tr>
<tr>
<td>5</td>
<td>40</td>
<td></td>
<td>total_iters</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 1</td>
</tr>
<tr>
<td>6</td>
<td>1057</td>
<td></td>
<td>total_iters</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 2</td>
</tr>
</tbody>
</table>
### Metadata Collection for Performance Analysis

Karen L. Karavanic  July 17, 2007

#### Selection Parameters

<table>
<thead>
<tr>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>execution/PT.su2_moi-158</td>
<td></td>
</tr>
</tbody>
</table>

#### Table

<table>
<thead>
<tr>
<th>saved</th>
<th>value</th>
<th>units</th>
<th>metric</th>
<th>label</th>
<th>combined</th>
<th>start</th>
<th>end</th>
<th>time/interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>46.24673</td>
<td>seconds</td>
<td>/Time</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 1</td>
</tr>
<tr>
<td>2</td>
<td>55.01562</td>
<td>seconds</td>
<td>/Time</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 2</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td></td>
<td>Average cg iters for s</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 1</td>
</tr>
<tr>
<td>4</td>
<td>176</td>
<td></td>
<td>Average cg iters for s</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 2</td>
</tr>
<tr>
<td>5</td>
<td>40</td>
<td></td>
<td>Total_iters</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 1</td>
</tr>
<tr>
<td>6</td>
<td>1057</td>
<td></td>
<td>Total_iters</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>/whole execution/main loop iteration 2</td>
</tr>
</tbody>
</table>

#### Select Operator

Select an operator from the list.

- sum

Buttons:
- OK
- Cancel
Metadata Collection for Performance Analysis

Karen L. Karavanic
July 17, 2007
### Metadata Collection for Performance Analysis

**Karen L. Karavanic**

July 17, 2007

---

**Selection Parameters**

<table>
<thead>
<tr>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>execution:PT_sw3_rmd-158</td>
<td></td>
</tr>
</tbody>
</table>

---

**Metrics Table**

<table>
<thead>
<tr>
<th>saved</th>
<th>value</th>
<th>units</th>
<th>metric</th>
<th>label</th>
<th>combined</th>
<th>start_time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td></td>
<td>Average cg iters for s</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>48.84073</td>
<td>seconds</td>
<td>/Time</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>40</td>
<td></td>
<td>Total_iters</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1.76</td>
<td></td>
<td>Average cg iters for s</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>55.01562</td>
<td>seconds</td>
<td>/Time</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1057</td>
<td></td>
<td>Total_iters</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>103.85635</td>
<td>seconds</td>
<td>/Total main loop Time: sw3_rmd-158-ppnl-r1</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
### Combined Performance Results

<table>
<thead>
<tr>
<th>Operator Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>sum</td>
<td>105.325</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Metrics Combined</th>
<th>Number of Values Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>/Time</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>New Metric Name</th>
<th>Units</th>
<th>Start Time</th>
<th>End Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>/Total main loop Time</td>
<td>seconds</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Label</td>
<td>su3_rmd-170-ppn2-n8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>saved</td>
<td>value</td>
<td>units</td>
<td>combined</td>
</tr>
<tr>
<td>-------</td>
<td>-----------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>1</td>
<td>103.35635</td>
<td>seconds</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>105.32174</td>
<td>seconds</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>103.29681</td>
<td>seconds</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>112.45473</td>
<td>seconds</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>113.30979</td>
<td>seconds</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>117.36985</td>
<td>seconds</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>114.48338</td>
<td>seconds</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>116.31307</td>
<td>seconds</td>
<td></td>
</tr>
</tbody>
</table>
"Metadata Collection for Performance Analysis"
Karen L. Karavanic July 17, 2007
How do people want to use PerfTrack?

- Traditional Performance Analysis/Tuning
- Fully automated performance regression testing
- Comparative evaluation of new platforms vs. old
- Effects of hardware and software upgrades
- OS kernel performance study
- Organized store to replace scattered files
- Sharing a single data store in collaborative studies
What do people want to store in PerfTrack?

- All Performance Experiment Artifacts
  - HW counter data, profile data, trace data, benchmark output
  - "Barry’s World": create the graph, save the graph, save the steps to create the graph
  - Paradyn artifacts: Search History Graph, Call graph, performance data histograms
  - Data from all common tools -- OpenSpeedshop, TAU, etc.
  - As much description of the build and runtime environments as possible
Machine Data Collection

- "Automated System Environment Capture For PerfTrack"
  Capstone Project Team: Aaron Amauba, Dave Vu, Steve Wooster
- What to collect?
- When to collect?
- We already have this information... right??
- Who knows?
- Device model number example
- Is the execution timestamp enough?
Machine Data Collection: Host System method

- **Idea**
  - Write scripts to run on the host system and directly measure the environment

- **Features**
  - Modular structure. User can specify tailor made modules for the system they are interested in.
  - Can be run anytime. Just kick off the script.
  - Allow the user to provide resource hierarchy information when a connection to the database is not available.

- **Status**
  - Currently implemented and tested for Linux
Machine Description -- Scaling Challenges

• Automated Host Data Collection eliminates tedious manual entry
  BUT
  • When to scan? BG/L -- 100k+ nodes; update frequency ??
    - Per Execution?? Weekly? Daily? Hourly?
  • Who scans?
    - Each researcher ?
      • (requires running a "scan" program on each node, can be difficult to get these types of jobs scheduled by scheduler)
      • Do we really need all those copies??
  • Delay in entry -> overwrite new data with old?? names??
  • The best answer will involve lab support!
Collaborative Data Stores

- PERI-DB (Shirley Moore)
- Goal: Develop and deploy a data store for performance data sets of PERI project researchers
- Approach
  - Define PERI XML schema
  - Tools provide a mapping to/from PERI XML
- We extended our data collection scripts to output PERI xml
- In progress: translating PTdf <=> PERI xml
  - conversion of PTdf files to PERI xml
  - conversion of PERI xml files to PTdf
  - input of PERI xml to PerfTrack database
  - export of PERI xml from queries to PerfTrack database
Key Issues in Metadata Collection

• Rich Data Sets
  - build, platform, runtime environment, performance data
  - sparse data will impact results -- eg clustering

• Scalability
  - collection frequency: each run? each experiment? each user? each boot? each upgrade?
  - attributes for groups of resources
  - Time: when is new knowledge created?
  - new machine resource every year
  - attributes of machine resource change every few weeks
  - input data changes every few runs
  - runtime environment may change during one execution
  - diagnoses and comparative data during analysis
Key Issues in Metadata Collection

- **Collaborative Data Stores**
  - Need to map resources and results between local sites

- **Porting difficulties**
  - Commands for collecting metadata information vary from platform to platform

- **Lack of common interfaces**
  - e.g. file system software version information on Linux: Lustre vs GPFS
The PerfTrack Project

- http://www.llnl.gov/casc/perfTrack/
- http://www.cs.pdx.edu/~karavan/perftrack
- Karen Karavanic: karavan@cs.pdx.edu  John May: johnmay@llnl.gov
- Kathryn Mohror, Rashawn Knapp, Nagalaxmi Karumbunathan
- This research supported in part by UC/LLNL subcontract #B539302.
- Portions of this work were performed under the auspices of the U.S. Department of Energy by the University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.
- Clustering interfaces: Thomas Conerly, Abraham Neben (Portland Saturday Academy Internship Program for high school students)
- PPerfGrid: John Hoffman (PSU masters thesis)
- Capstone Project: Aaron Amauba, Dave Vu, Steve Wooster (PSU undergraduate course)