Benchmark Precision and Random Initial State

Tomas Kalibera, Lubomir Bulej, Petr Tuma

DISTRIBUTED SYSTEMS RESEARCH GROUP
http://nenya.ms.mff.cuni.cz

CHARLES UNIVERSITY PRAGUE
Faculty of Mathematics and Physics
Goal: Tool for improving software performance.

- Regular automatic benchmarking
  - Incorporate into regression testing
- Automated detection of regressions
  - Detect changes in benchmark results
- Fixing important regressions
  - Automatically find suspect modifications
  - (Manually) fix regressions if possible

http://nenya.ms.mff.cuni.cz/projects/mono
Proceedings: pg. 853
Benchmarks are unstable.
Benchmark results differ in each execution.

Individual samples, vertical lines denote new runs.
Random state is integral part of real systems.

- Differences in results from different executions cannot be removed by
  - Shutting down non-related services
  - Disconnecting network, unloading drivers
  - Turning off randomization of virtual addresses
  - Rebooting before each benchmark execution
  - Excessively long warm-up phase in each execution
The problem can be quantified.

- Impact factor of random initial state
  - Robust to non-normality, outliers
  - Calculated from benchmark results by simple statistical simulation
- Defined as ratio of variability in data from different runs to variability in data from the same run
  - Values $\geq 1$, 1 means no impact
Impact of random state is system dependent.

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Platform</th>
<th>Impact Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFT</td>
<td>Pentium/Windows</td>
<td>94.74</td>
</tr>
<tr>
<td>FFT</td>
<td>Itanium/Linux</td>
<td>35.91</td>
</tr>
<tr>
<td>FFT</td>
<td>Pentium/Linux</td>
<td>25.81</td>
</tr>
<tr>
<td>FFT</td>
<td>Pentium/DOS</td>
<td>1.06</td>
</tr>
<tr>
<td>RPC Marshaling</td>
<td>Pentium/Linux</td>
<td>2.61</td>
</tr>
<tr>
<td>RPC Ping</td>
<td>Pentium/Linux</td>
<td>1.10</td>
</tr>
<tr>
<td>RUBiS</td>
<td>Pentium/Linux</td>
<td>1.01</td>
</tr>
</tbody>
</table>
Differences in results are due to cache misses.
**Conclusion: Benchmarking is still possible.**

- Random initial state is a reality
- Implications for benchmarking
  - Need to run more times, possibly re-compile
  - Non-trivial statistical evaluation required
- Current status
  - Simple hierarchical model
  - Allows precision estimation, experiment planning
Regression benchmarking publications

- Kalibera, T., Bulej, L., Tuma, P.: **Quality Assurance in Performance: Evaluating Mono Benchmark Results**, accepted as a full paper on Second International Workshop on Software Quality (SOQUA 2005), Erfurt, Germany