Performance Awareness

Keynote Abstract

Petr Tůma
Department of Distributed and Dependable Systems
Faculty of Mathematics and Physics
Charles University, Prague, Czech Republic
petr.tuma@d3s.mff.cuni.cz

ABSTRACT

The talk will take a broad look at performance awareness, defined as the ability to observe performance and to act on the observations. The implicit question posed in the talk is what can be done to improve various aspects of performance awareness – be it our awareness of the various performance relevant mechanisms, our awareness of the expected software performance, our ability to attain and exploit performance awareness as software developers, and our options for implementing performance aware applications.

Keywords

Performance Awareness, Development, Documentation, Measurement

Categories and Subject Descriptors

D.4.8 [**Performance**]: Measurements

General Terms

Performance

1. OVERVIEW

In broad terms, performance awareness can be defined as the ability to observe performance and to act on the observations. Performance awareness should permeate software development – at many levels, from architectural design through implementation to eventual maintenance, decisions need to be made that balance performance against factors such as development effort or maintenance cost – and without performance awareness, this balance cannot be achieved.

Performance awareness is often gained through experimental performance evaluation. Observation of live systems is used to learn about actual performance and to discover and analyze potential performance anomalies ; experimental benchmarks of various complexity are used to evaluate software and hardware designs ; theoretical performance models are often validated against experimental measurements.

Because thorough experimental performance evaluation can be both difficult and expensive, software development may rely on an evaluation that is accidentally or intentionally limited. A limited evaluation may contribute to incomplete awareness, which in turn increases the potential for less-than-ideal development decisions that yield less-than-optimal software systems.

Using examples of performance evaluation experiments, the talk will argue that opportunities for relatively simple improvements exist in the methods and tools we use to achieve performance awareness. The talk structure will look at four performance awareness topics:

Mechanisms. Contemporary systems include complex performance relevant mechanisms that interact to determine the observed performance. Performance awareness requires learning about these mechanisms in an efficient manner.

Expectations. The observed performance of a system is a result of both deliberate design and accidental interactions. Compared to awareness of the observed performance, awareness of the design intent can be more convenient in some software development tasks.

Developers. It is difficult to anticipate which steps in the software development process will significantly influence system performance. Besides working on automated optimizations that tend to hide performance relevant mechanisms, we should work on efficient methods and tools that provide developers with performance awareness to complement the optimizations.

Applications. Dynamic nature of performance often requires adaptive applications that possess performance awareness. Despite this need, performance awareness is still not treated the same as other forms of software reflection.

ICPE'14, March 22–26, 2014, Dublin, Ireland
ACM 978-1-4503-2733-6/14/03.
http://dx.doi.org/10.1145/2568088.2576097.

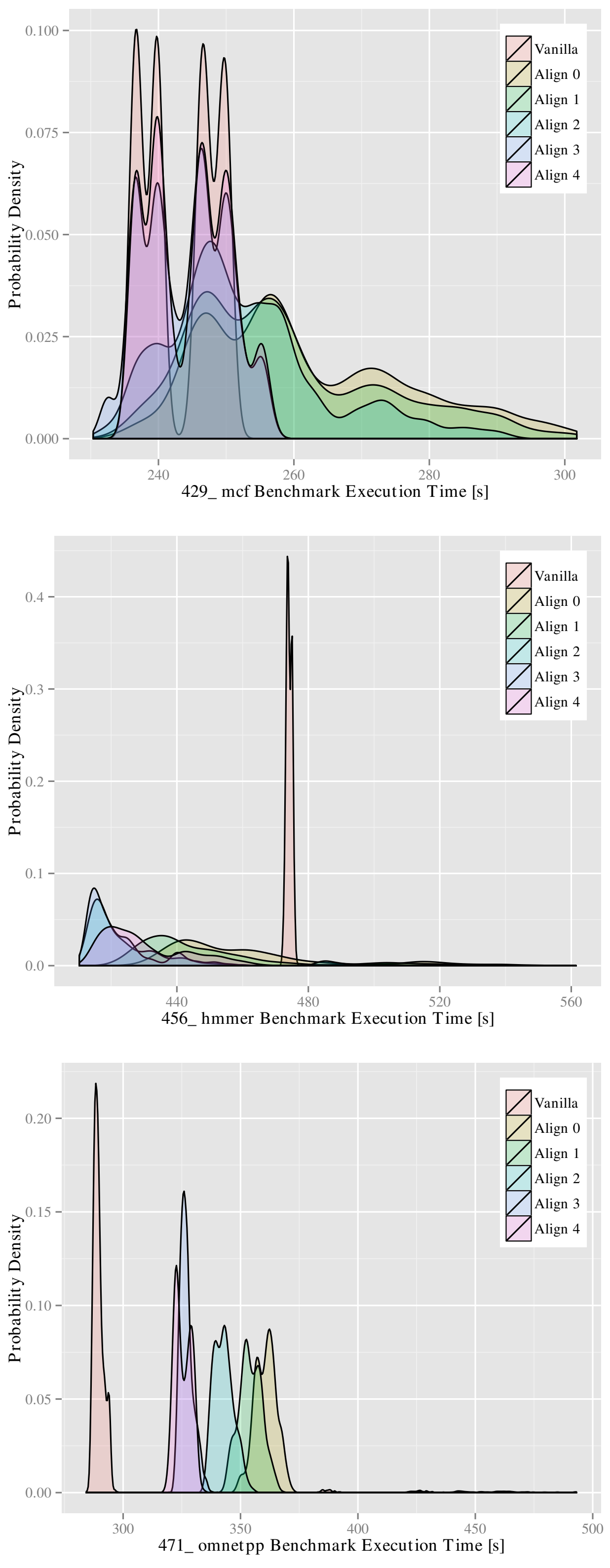

Figure 1: Allocation alignment impact.

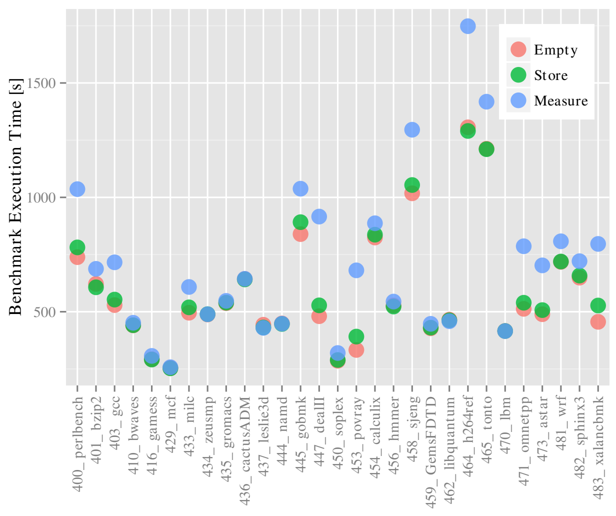

Figure 2: Instrumentation overhead.

2. MEASUREMENTS

The keynote abstract reproduces several of the experimental measurements from the talk for better readability. Figure 1 illustrates the impact of heap object placement on the SPEC CPU2006 results. Figure 2 shows the overhead of instrumenting all exported functions of SPEC CPU2006 with timestamping. The measurements were conducted on a server with two Intel Xeon E5-2660 processors, 48 GB RAM, running the latest packages from Fedora 18 and Fedora 20 distributions of Linux.

3. ACKNOWLEDGMENTS

Collaboration with my current and former colleagues, for which I remain ever grateful, has naturally contributed to the presented results. The EU and GAČR funding was essential for much of our activities.