SPEC CPU2006 Analysis Papers: Guest Editor's Introduction

John L. Henning, Secretary, SPEC CPU Subcommittee
john dot henning at acm dot org

During the development of the new benchmark suite CPU2006, SPEC analyzed benchmark candidates for various technical attributes, including time profiles, language standard compliance, I/O activity, system resource usage, and many other attributes. Many people contributed to the analysis, as shown in the credits at www.spec.org/cpu2006/docs/credits.html. This issue of Computer Architecture News presents a set of articles flowing from that analysis effort.

Three papers discuss the benchmarks that have been selected for the suite:

- **"SPEC CPU Suite Growth: An Historical Perspective"** by John L. Henning examines how the SPEC CPU suites have grown over the years, and discusses some of the reasons for that growth.
- Using a set of hardware performance counters, Aashish Phansalkar, Ajay Joshi and Lizy K. John consider similarities among the benchmarks for their article **"Subsetting the SPEC CPU2006 Benchmark Suite"**. The intent is to assist users of performance simulators who, because of time constraints, prefer to simulate a representative subset of the benchmarks. Note: although SPEC is aware that users of performance simulators may prefer to study subsets, SPEC does not endorse any subset as representative of the overall metrics.
- The CPU2006 suite includes far more C++ content than previous CPU suites. Michael Wong reviews C++ language and standards issues that have affected the benchmark selection and porting process, with his article **"C++ Benchmarks in SPEC CPU2006"**.

Three papers present information about the new suite's memory behavior:

- **"SPEC CPU2006 Memory Footprint"** by John L. Henning graphs virtual and physical memory consumption over time using the traditional Unix metrics of rss (resident set size) and vsz (virtual size).
- In **"CPU2006 Working Set Size"**, Darryl Gove takes a closer look at memory use by using the SHADE instruction analyzer to track load and store instructions. The article provides two metrics that track memory which is actively used - a much smaller amount than the traditional rss or vsz - and compares CPU2000 vs. CPU2006.
- Because SPEC CPU2006 includes codes with a larger memory footprint, there is substantial runtime variation depending on the selection of memory page sizes. Wendy Korn and Moon S. Chang show runtime effects from 4KB, 64KB, and 16MB pages in their article **"SPEC CPU2006 Sensitivity to Memory Page Sizes"**.

Four papers examine technical behavior of the benchmarks in the light of SPEC's goals for them:

- The beginning point for analysis of program performance behavior is to profile where the time went. Reinhold P. Weicker and John L. Henning provide that beginning in **"Subroutine Profiling Results for the CPU2006 Benchmarks"**. They discuss SPEC's goals for usage of library routines, and how those goals were adjusted as the subcommittee learned more about the application areas represented in CPU2006.
- For the CPU suites, SPEC seeks the compute-intensive portion of applications, not the I/O intensive portion. By the time SPEC completed its work, were the benchmarks compute bound, with little I/O? Dong Ye, Joydeep Ray, and David Kaeli answer that question in their **"Characterization of File I/O Activity for SPEC CPU2006"**.
- Hardware performance counters can provide insight into benchmark candidates. John L. Henning discusses SPEC's use of counters in the paper **"Performance Counters and Development of SPEC CPU2006"**.
- SPEC allows compilation with Feedback Directed Optimization (FDO), also known as Profile Based Optimization. (For CPU2006, FDO is permitted in peak; CPU2000 permitted it in both base and peak.) Sometimes, it is hard to find a useful FDO training data set. In their article **"Evaluating the correspondence between training and reference workloads in SPEC CPU2006"**, Darryl Gove and Lawrence Spracklen examine whether training workloads visit the same instructions as reference workloads, and whether branches behave similarly.

Finally, SPEC's toolset is central to providing a controlled environment for running the benchmarks with documented options and clear reporting. Cloyce D. Spradling provides an overview of the toolset, including how to use its hooks for analysis tools, in his article **"SPEC CPU2006 Benchmark Tools"**.

Since many of the articles that follow reference various SPEC CPU2006 benchmarks, for convenience, the next page summarizes them. More detailed descriptions were published in Computer Architecture News, Volume 34, number 4, also posted at www.spec.org/cpu2006.

[If you are reading this document in a PDF viewer, you may be able to click article titles above.]

CINT2006 (Integer Component of SPEC CPU2006):

Benchmark	Lan-guage	Application Area	Brief Description
400.perlbench	C	Programming Language	Derived from Perl V5.8.7. Workload includes SpamAssassin, MHonArc (an email indexer), and specdiff (SPEC's tool to check benchmark outputs).
401.bzip2	C	Compression	Julian Seward's bzip2 version 1.0.3, modified to do most work in memory, rather than doing I/O.
403.gcc	C	C Compiler	Based on gcc Version 3.2, generates code for Opteron.
429.mcf	C	Combinatorial Optimization	Vehicle scheduling. Uses a network simplex algorithm (which is also used in commercial products) to schedule public transport.
445.gobmk	C	Artificial Intelligence: Go	Plays the game of Go, a simply described but deeply complex game.
456.hmmer	C	Search Gene Sequence	Protein sequence analysis using profile hidden Markov models
458.sjeng	C	AI: chess	A highly-ranked chess program that also plays several chess variants.
462.libquantum	C	Physics/Quantum Comp.	Simulates quantum computer; runs Shor's polynomial-time factorization.
464.h264ref	C	Video Compression	A reference implementation of H.264/AVC, encodes a videostream using 2 parameter sets. The H.264/AVC standard is expected to replace MPEG2
471.omnetpp	C++	Discrete Event Simulation	Uses the OMNet++ discrete event simulator to model a campus network.
473.astar	C++	Path-finding Algorithms	Pathfinding library for 2D maps, including the well known A* algorithm.
483.xalancbmk	C++	XML Processing	Transforms XML documents to other docs using a modified Xalan-C++

CFP2006 (Floating Point Component of SPEC CPU2006):

Benchmark	Lan-guage	Application Area	Brief Description
410.bwaves	Fortran	Fluid Dynamics	Computes 3D transonic transient laminar viscous flow.
416.gamess	Fortran	Quantum Chemistry.	Implements a wide range of quantum chemical computations. Test case does self-consistent field calculations using several methods.
433.milc	C	Physics/Chromodynamics	A gauge field program: lattice gauge theory with dynamical quarks.
434.zeusmp	Fortran	Physics / CFD	A computational fluid dynamics code developed at NCSA for simulation of astrophysical phenomena.
435.gromacs	C,Fortran	Biochem/Molecular Dyn.	Molecular dynamics: simulate Newtonian equations of motion The test case simulates protein Lysozyme in a solution.
436.cactusADM	C,Fortran	Physics / Relativity	Einstein evolution equations, staggered-leapfrog numerical method
437.leslie3d	Fortran	Fluid Dynamics	Computational Fluid Dynamics (CFD) using Large-Eddy Simulations. Uses MacCormack Predictor-Corrector time integration
444.namd	C++	Biochem/Molecular Dyn..	Simulates biomolecular system: 92,224 atoms of apolipoprotein A-I.
447.dealII	C++	Finite Element Analysis	C++ library for adaptive finite elements and error estimation. Test case is a Helmholtz-type equation with non-constant coefficients.
450.soplex	C++	Linear Programming, Optimization	Solves a linear program using a simplex algorithm and sparse linear algebra. Test cases incl. railroad planning and military airlift models.
453.povray	C++	Image Ray-tracing	Image rendering. Test case is a 1280x1024 anti-aliased image with some abstract objects with textures using a Perlin noise function.
454.calculix	C,Fortran	Structural Mechanics	Finite element code. Uses the SPOOLES solver library.
459.GemsFDTD	Fortran	Electromagnetics	Maxwell equations in 3D, finite-difference time-domain method.
465.tonto	Fortran	Quantum Chemistry	An open source quantum chemistry package, using an object-oriented design in Fortran 95. Test case places a constraint on a molecular Hartree-Fock wave function calculation.
470.lbm	C	Fluid Dynamics	Lattice-Boltzmann Method to simulate incompressible fluids in 3D
481.wrf	C,Fortran	Weather	Weather modeling. The test case is from a 30km area over 2 days.
482.sphinx3	C	Speech recognition	Carnegie Mellon University's well-known speech recognition system.

[If you are reading this document in a PDF viewer, you may be able to click benchmark names above for more detail.]