CMP SCI 635: Modern Computer Architecture

Professor Charles Weems

Analysis of Redundancy and Application Balance in the SPEC CPU2006 Benchmark Suite

Name: Kunjal Panchal Date: 12th Sept, 2019

Student ID: 32126469 Paper: 3 – Methodology [John06]

Strengths:

1. SPEC2006 is old, it is considered a really good test because it takes over an hour, and many people have deemed the work done as "practical" and "representative" of real programs. In fact, they're really just real programs that were used back in 2006 very often. So, from 2006 standards point of view, this was efficient, considering the variety it provided e.g. Chess AI.

2. The Principle Component Analysis or Cluster Analysis techniques used in this paper shows quite commendable results. CINT and CFP subsets are almost providing same aggregate results. Proving that SPEC2006 suite has redundant programs in it and thus a researcher need not utilize all the program individually under some time constraints.

3. They made subsets not just from behavioural characteristics of benchmarks but also from structural characteristics i.e. PCA/CA are behavioural, while subsets based on branch and memory access characteristics are structural. Leaving us with the choice of what kind of program we have for evaluation.

4. {Meta} Prompted me to discover and read about SPEC CPU2017 benchmarks.

Weaknesses:

1. {As compared to the newly released SPEC2017}
   https://www.spec.org/cpu2017/Docs/overview.html#benchmarks
   There's a statement which goes like this: "*Caution: The benchmarks do not comply perfectly with ISO/ANSI lanaguage standards, because their source code is derived from real applications. The rules allow optimizers to assume standards *only* where that does not prevent validation.*"
   Which is same as in SPEC2006.
   Clearly, the compiler is allowed to recognize that SPEC is being compiled and just print the expected results.
   SPEC2006 has a number of test cases with "*undefined behaviour*".

2. SPEC2006, which has overall run for longer and has longer tests, will much better represent the sustained (single-core) CPU performance of whatever device you're running it on. However, that still doesn't mean that a. the workload you care about behaves similarly to the tests that are in SPEC2006 or that b. your workload is primarily CPU-bound.

3. From an article [https://techreport.com/news/32126/spec-updates-its-cpu-benchmark-for-the-first-time-in-11-years/] on SPEC2017,
   "*The new tests eliminate the libquantum test component, which was believed to give Intel processors an advantage over the competition. The Intel C++ Compiler had "cracked" libquantum, which may have inflated overall benchmark scores by 5-10% on Intel CPUs.*"
   Implying that Intel's compiler has been cheating in this CPU benchmark all these years.

4. The benchmark *libquantum* was always considered an awkward outlier and many researchers couldn't figure out it's behaviour.

5. The researchers performed PCA manually, given that it's not impossible, analysing

   (5 different architectures) * (6 * characteristics per program) * (29 programs on each architecture) = 870 characteristics

   Which might easily get out of hand and there are chances of human errors.

6. The author finds it redundant to have more than one benchmark from the same field but looking through one less layer of abstraction, we find that even if *458.sjeng* and *473.astar* are AI programs, they are Chess Playing and Path Finding algorithms respectively, which can be totally different in logic, structure and behaviour. Same argument applies to *410.bwaves* and *437.leslied*

Questions/Assertions:

1. How does SPEC choose the programs for its benchmark suite? Same question for input test data suite. Do they also perform PCA/CA for all the licensed programs and choose the best one that summarizes a category? Why doesn't SPEC performs similarity search before releasing the benchmark and remove all those are too similar for their user community?

2. Does an input set and a benchmark suite have any co-relation between them? How valid is it to use one particular program's input for another very similar, but not the same, program? Aren't inputs in any particular format which other programs might not be adhering to or they might not recognize the format.