How to Use SPEC Benchmarks for Decision Making

Sunita Chandrasekaran, Robert Henschel, Junjie Li, Veronica G. Vergara Larrea

SPEC HPG: https://www.spec.org/hpg
Tutorial Website: https://www.olcf.ornl.gov/sc20-spec-hpg-tutorial
I. SPEC benchmark philosophy
   a) Structure of SPEC benchmarks
      • components & workloads & tools
      • SPEC score
      • brief story of run rules
      • Configuration files
      • From base to peak runs
   b) Result disclosure
      • Documentation and benchmark report
      • Reportable runs
      • the peer-review process for publishing

II. Use SPEC benchmarks for decision making
   • procurement process
   • test performance of compilers
   • compiler validation
   • explore performance of new architectures

III. Benchmark acquisition & licensing
   • commercial license
   • academic license
   • become a member
I. SPEC benchmark philosophy

a) Structure of SPEC benchmarks
   • components & workloads & tools
   • SPEC score
   • brief story of run rules
   • Configuration files
   • From base to peak runs

b) Result disclosure
   • Documentation and benchmark report
   • Reportable runs
   • the peer-review process for publishing

II. Use SPEC benchmarks for decision making

   § procurement process
   § test performance of compilers
   § compiler validation
   § explore performance of new architectures

III. Benchmark acquisition & licensing

   § commercial license
   § academic license
   § become a member
Benchmark components

SPEC benchmarks contain:

- benchmark codes
- benchmark workloads (input and output)
- SPEC tools
- run rule (every benchmark is different)
**SPEC Accel Benchmark Suites**

**Benchmark components**

**SPEC Accel suites:**
- 34 benchmark codes in 3 suites

**programming mode suites**

- **OpenACC**
  - 15 codes

- **OpenMP target offload**
  - Same 15 codes with different directives

- **OpenCL**
  - 19 codes

**data set**

- **test** (quickly test your binary)
- **train** (for feedback-directed optimization)
- **ref** (for production run and reporting)

---

**results from different suites are not comparable**

---

**Junjies-MBP16:ACCEL junjies\$ ls**

<table>
<thead>
<tr>
<th>benchmark</th>
<th>test</th>
<th>train</th>
<th>ref</th>
</tr>
</thead>
</table>

---

SC20 SPEC Tutorial, Part C: How to use SPEC benchmarks for decision making
## Benchmark components

### SPEC Accel Benchmark Suites

<table>
<thead>
<tr>
<th>OpenACC/OpenMP4, 5 Benchmarks</th>
<th>Language</th>
<th>Origin</th>
<th>Application Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ostencil</td>
<td>C</td>
<td>Parboil, University of Illinois</td>
<td>Thermodynamics</td>
</tr>
<tr>
<td>olbm</td>
<td>C</td>
<td>Parboil, University of Illinois, SPEC CPU2006</td>
<td>Computational Fluid Dynamics, Lattice Boltzmann</td>
</tr>
<tr>
<td>omriq</td>
<td>C</td>
<td>Rodinia, University of Virginia</td>
<td>Medicine</td>
</tr>
<tr>
<td>md</td>
<td>Fortran</td>
<td>Indiana University</td>
<td>Molecular Dynamics</td>
</tr>
<tr>
<td>palm</td>
<td>Fortran</td>
<td>Leibniz University of Hannover</td>
<td>Large-eddy simulation, atmospheric turbulence</td>
</tr>
<tr>
<td>ep</td>
<td>C</td>
<td>NAS Parallel Benchmarks</td>
<td>Embarrassingly Parallel</td>
</tr>
<tr>
<td>clvleaf</td>
<td>C, Fortran</td>
<td>Atomic Weapons Establishment</td>
<td>Explicit Hydrodynamics</td>
</tr>
<tr>
<td>cg</td>
<td>C</td>
<td>NAS Parallel Benchmarks</td>
<td>Conjugate Gradient Solver</td>
</tr>
<tr>
<td>seismic</td>
<td>Fortran</td>
<td>GeoDynamics.org, University of Pau</td>
<td>Seismic Wave Modeling (PDE)</td>
</tr>
<tr>
<td>sp</td>
<td>Fortran</td>
<td>NAS Parallel Benchmarks</td>
<td>Scalar Penta-diagonal solver</td>
</tr>
<tr>
<td>csp</td>
<td>C</td>
<td>NAS Parallel Benchmarks</td>
<td>Scalar Penta-diagonal solver</td>
</tr>
<tr>
<td>miniGhost</td>
<td>C, Fortran</td>
<td>Sandia National Lab</td>
<td>Finite difference</td>
</tr>
<tr>
<td>ilbdc</td>
<td>Fortran</td>
<td>SPEC OMP2012</td>
<td>Fluid Mechanics</td>
</tr>
<tr>
<td>swim</td>
<td>Fortran</td>
<td>SPEC OMP2012</td>
<td>Weather</td>
</tr>
<tr>
<td>bt</td>
<td>C</td>
<td>NAS Parallel Benchmarks</td>
<td>Block Triangular Solver for 3D PDE</td>
</tr>
</tbody>
</table>

### SPEC Accel suites:
- the 15 codes in OpenACC suite
- real app or mini-app
- same codes later ported to OpenMP target offload suite
- coverages different scientific domains, different programming languages
Benchmark components

SPEChpc 2021 Benchmark Suites

workload

- small (scale up to 512 ranks)
- medium (scale up to ~8000 ranks)
- large (under development)

data set

- test (quickly test your binary)
- ref (for production run)

SPEChpc 2021 suites:
- 9~12 benchmark codes
- all codes support MPI, MPI+OpenMP, MPI+OpenACC, MPI+OpenMP target offload

For latest development of SPEChpc 2021
https://www.spec.org/hpc2021/
Benchmark components

- Codes are characterized during the selection process
- Coming from real-world applications
- May have completely different characteristics
- Identification: number and name
  - 350.md for example
- Documentation: website/ benchmark tree
  - Language, application domain,…

### Intel Vtune Amplifier Results

<table>
<thead>
<tr>
<th></th>
<th>350.md</th>
<th>363.swim</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPI rate</td>
<td>0.698</td>
<td>1.634</td>
</tr>
<tr>
<td>CPU utilization</td>
<td>69 %</td>
<td>23.6 %</td>
</tr>
<tr>
<td>Memory bound</td>
<td>11.8 %</td>
<td>48.5 %</td>
</tr>
<tr>
<td>% of packed FP instr</td>
<td>9.7 %</td>
<td>100 %</td>
</tr>
</tbody>
</table>

See SPEC HPG publication page for more performance studies:
https://www.spec.org/hpg/publications/

Intel Vtune Amplifier results @ 2x Intel Broadwell EP w/ 24 cores
Tools provided to ensure consistent operation of benchmarks across variety of platforms

- **runspec or runhpc**
  - Primary tool in the suite
  - *build, run, report all through this command* (you don’t manually run each code).
  - **Config file** needed for usage (with detailed instructions on building/running the benchmarks)

- **rawformat**
  - Results formatter needed for publishing SPEC results

- **specmake, specperl, specssha512sum, specdiff, specpp, specrand, and many more**
  - make, perl, sha512sum, etc.

Config Files

- Contain instructions
  - Building benchmarks
  - Running them
  - Description of system under test

- How to write a config file?
  - Often start off using a config file that someone else has previously written\(^1\)
    - example config in $SPEC/config/
    - from SPEC result submissions similar to your system\(^2\)
  - Write your own\(^3\)

Key for reproducibility!

---

<table>
<thead>
<tr>
<th>Test Sponsor</th>
<th>System Name</th>
<th>Base Threads</th>
<th>Enabled</th>
<th>Enabled Cores</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel</td>
<td>Intel Server</td>
<td>112</td>
<td>56</td>
<td>2</td>
<td>21.1</td>
</tr>
<tr>
<td></td>
<td>System R2208WFTZS (2 x Intel Xeon Platinum 8180, DDR4-2666, SMT ON Turbo ON)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lenovo Global</td>
<td>ThinkSystem</td>
<td>224</td>
<td>112</td>
<td>4</td>
<td>40.2</td>
</tr>
<tr>
<td>Technology</td>
<td>SR930</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle Corporation</td>
<td>SPARC T7-4</td>
<td>256</td>
<td>128</td>
<td>4</td>
<td>26.4</td>
</tr>
<tr>
<td>RWTH University</td>
<td>NEC HPC 1812Rg</td>
<td>24</td>
<td>24</td>
<td>2</td>
<td>7.22</td>
</tr>
<tr>
<td>Aachen</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RWTH University</td>
<td>NEC HPC 1812Rg</td>
<td>48</td>
<td>24</td>
<td>2</td>
<td>7.65</td>
</tr>
<tr>
<td>Aachen</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RWTH University</td>
<td>NEC HPC 1812Rg</td>
<td>12</td>
<td>144</td>
<td>8</td>
<td>33.6</td>
</tr>
<tr>
<td>Aachen</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RWTH University</td>
<td>NEC HPC 1812Rg</td>
<td>72</td>
<td>144</td>
<td>8</td>
<td>19.5</td>
</tr>
<tr>
<td>Aachen</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RWTH University</td>
<td>NEC HPC 1812Rg</td>
<td>36</td>
<td>144</td>
<td>8</td>
<td>9.97</td>
</tr>
<tr>
<td>Aachen</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RWTH University</td>
<td>NEC HPC 1812Rg</td>
<td>18</td>
<td>144</td>
<td>8</td>
<td>5.23</td>
</tr>
<tr>
<td>Aachen</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Source: as of May 2018\(^2\)

---

1. [https://www.spec.org/accel/docs/runspec.html#about_config](https://www.spec.org/accel/docs/runspec.html#about_config)
3. [https://www.spec.org/accel/docs/config.html](https://www.spec.org/accel/docs/config.html)
### Config Files

**Header section**
- Usually runspec flags

```plaintext
# what to do: build, validate = build+ run+ check+ report
action = validate
# Number of iterations of a test
iterations = 1
# Tuning levels: base, peak
tune = base
# Dataset size: test, train, ref
size = test
# Environment variable will be set using "ENV_*"
env_vars = 1
# Output format: all, pdf, text, html and so on
output_format = text
# The variable "command" is the command used by spec
tweak = yes
# Run benchmarks according to your specific system config
# The variable "command" is the command used by spec
# submit = <system command> $command
submit = <system command> $command
```

**Compiler Specification**

- **Compiler Flags**
  - Base & Peak (optional)
  - Portability

```plaintext
# Base
openacc = base=default=default:
OPTIMIZE = -fast -Mfile
PREFER_SIZE = -acc -ta=teslaacc35,cuda5.5
COPTIMIZE = -acc -ta=teslaacc35,cuda5.5

# Portability flags for each benchmark
# Following flag should not have any impact on performance.
# This information will be extracted for a reportable run.

359,miniGhost=default=default=default:
EXTRA_LDFLAGS += -ta=tesla:cc35
```

**Host Information**

```plaintext
# # HOST Hardware information
# # Copyright (C) 2012-2014 SPEC
# default=default=default=default:
hw_avail = Nov-2013
hw_cpu_name = Intel Core i7-3930K
```

**Accelerator Information**

```plaintext
# # Accelerator Hardware information
# #copyright (C) 2012-2014 SPEC
# hw_accel_desc000 = GPU Boost set to use a graphic clock frequency
# hw_accel_desc001 = of 810 MHz. See notes below.
```

**MD5 section**

- Automatically-generated

SC20 SPEC Tutorial, Part C: How to use SPEC benchmarks for decision making
SC20 SPEC Tutorial, Part C: How to use SPEC benchmarks for decision making

# Software information

# Compilers.
default=default=default=default:
CC = mpicc
CXX = mpic++
FC = mpifortran

# Base

openacc=base=default=default=default:
OPTIMIZE = -fast -Mfprelaxed
FOPTIMIZE = -acc -ta=tesla:cc70,cuda10.1
COPTIMIZE = -acc -ta=tesla:cc70,cuda10.1

- Settings for base runs
- same flag for all
- Compiler flags or environment variables
# Environment variables at runtime
ENV_PGI_ACC_BUFFERSIZE = 8M

# Portability flags for each benchmark
# Following flag should not have any impact on performance.

Set portability flags if benchmark cannot be built/execute correctly w/o these flags.

Other typical portability flags: language standards (-std=c99), Fortran format (-free), memory models (-mcmodel=medium)

SPEC Accel OpenACC current doesn’t need portability flag with PGI.

Environment variables
- Active if env_vars is set to 1 (see prior slides)
- Need to start with ENV_
# Hardware and software information for the machine under test.
# This information will be extracted for a reportable run.
# An example configuration can be copied from the website
# https://www.spec.org/accel/results/accel_acc.html

---

```plaintext
company_name = SPEC Tutorial Company
test_sponsor = SPEC Tutorial Sponsor
tester = SPEC Tutorial Tester
license_num = SPEC Tutorial License
machine_name = SPEC Tutorial Machine

hw_vendor = NEC
hw_avail = NOV-2016
hw_cpu = Intel Xeon E5-2650 v4
hw_cpu_mhz = 2200
hw_cpu_max_mhz = 2900
```
Config Files

# HOST Hardware information

default=default=default=default:

hw_avail = Nov-2013
hw_cpu_name = Intel Core i7-3930K
hw_cpu_mhz = 3200
hw_cpu_max_mhz = 3800
hw_fpu = Integrated
hw_nchips = 1
hw_ncores = 6
hw_ncoresperchip = 6
hw_nthreadspercore = 2
hw_ncpuorder = 1 chip
hw_pcache = 32 KB I + 32 KB D on chip per core
hw_scache = 256 KB I+D on chip per core
hw_tcache = 12 MB I+D on chip per chip

# Accelerator Hardware information

hw_accel_model = Tesla K40c
hw_accel_vendor = NVIDIA
hw_accel_name = NVIDIA Tesla K40c
hw_accel_type = GPU
hw_accel_connect = PCIe 3.0 16x
hw_accel_ecc = Yes

Performance critical aspects of Host and Accelerator are documented
# Software information

default=default=default=default:
    sw_avail = Feb-2014
    sw_compiler = PGI Accelerator Server Complete, Release 14.2
    sw_accel_driver = NVIDIA UNIX x86_64 Kernel Module 319.60

# MD5 section. It will be created by SPEC automatically.
# It is used by SPEC to check whether an executable is created using the current settings.

---

**Information on software configuration, e.g. compilers**

**MD5 section**
- Automatically generated by SPEC tools
- Used to check whether an executable is created using the current settings
SPEC Score

- Each benchmark component will have a performance “ratio”.
- “ratio”=speed up comparing to a reference machine (ratio of runtime)
- SPEC Score = geometric mean of “ratio” from all benchmark components
- SPEC Score of reference machine is “1”
- Higher is better, easy to understand

- Reference machines
  - For SPEC OMP2012
    - Sun Fire X4140, 2xAMD Opteron 2384, 8 cores, 2 chips, 4 cores/chip, 2.7 GHz
  - For SPEC Accel
    - SGI C3108-TY11, NVIDIA Tesla C2070, Intel Xeon E5620 2.4GHz
Run rules

fair, objective benchmarking and full reproducibility.

(full run rule is 20 page doc, only key points here)

- Most benchmarks forbids source code changes
  - rely on compiler and runtime for performance optimization
  - instead of testing programmer’s optimization skill, SPEC gauges what the software stack can do
  - SPEC Accel allows changes in directives for peak runs

OpenMP: https://www.spec.org/omp2012/docs/runrules.html
MPI: https://www.spec.org/mpi/docs/runrules.html
ACCEL: https://www.spec.org/accel/docs/runrules.html
Run rules

fair, objective benchmarking and full reproducibility.

(full run rule is 20 page doc, only key points here)

• Most benchmarks forbids source code changes
  § rely on compiler and runtime for performance optimization
  § instead of benchmarking programmer’s optimization skill, SPEC gauges what the software stack can do for regular users
  § SPEC Accel allows changes in directives for peak runs

• Base vs Peak:
  • Base run: must have the same compiler flags for all benchmark components
  • Peak run: compiler flags can be tuned for reach individual benchmark

OpenMP: https://www.spec.org/omp2012/docs/runrules.html
MPI: https://www.spec.org/mpi/docs/runrules.html
ACCEL: https://www.spec.org/accel/docs/runrules.html
Run rules

fair, objective benchmarking and full reproducibility. (full run rule is 20 page doc, only key points here)

• Most benchmarks forbids source code changes
  ▪ rely on compiler and runtime for performance optimization
  ▪ instead of benchmarking programmer’s optimization skill, SPEC gauges what the software stack can do for regular users
  ▪ SPEC Accel allows changes in directives for peak runs

• Base vs Peak:
  • Base run: must have the same compiler flags for all benchmark components
  • Peak run: compiler flags can be tuned for reach individual benchmark

• System under test needs to be fully documented for reproducibility

• Benchmark results are submitted to SPEC for peer review before going public (a two-week review period)

• All official SPEC results are published on SPEC website

OpenMP: https://www.spec.org/omp2012/docs/runrules.html
MPI: https://www.spec.org/mpi/docs/runrules.html
ACCEL: https://www.spec.org/accel/docs/runrules.html

SC20 SPEC Tutorial, Part C: How to use SPEC benchmarks for decision making
• Modifying the config file
  • Once you have a config file that runs on your system, it is easy to modify it
  • E.g. peak optimizations for better performance

• Example:

```
350.md: -O3 -openmp -ipo -xCORE-AVX2 -fno-alias -opt-malloc-options=1 -fp-model fast=2
         -no-prec-div -no-prec-sqrt -align array64byte

351.bwaves: -O3 -openmp -ipo -xCORE-AVX2 -fno-alias -fp-model fast=2 -no-prec-div -no-prec-sqrt
                -align array64byte

357.bt331: Same as 351.bwaves

360.ilbdc: basepeak = yes

362.fma3d: -O3 -openmp -ipo -xCORE-AVX2 -fno-alias -align array64byte

363.swim: -O3 -openmp -ipo -xCORE-AVX2 -fno-alias -opt-streaming-stores always
          -opt-malloc-options=3 -align array64byte

370.mgrid331: -O3 -openmp -ipo -xCORE-AVX2 -fno-alias -opt-malloc-options=3 -fp-model strict
```
Just show you how a peak run will look like:

```
# Peak
default=default=default:
OPTIMIZE    = -O3 -qopenmp -ipo -xCORE-AVX2 -no-prec-div
COPTIMIZE   = -ansi-alias
CXXOPTIMIZE = -ansi-alias
FOPTIMIZE   = -align

# [..] Environment variables

350.md=peak=default=default:
OPTIMIZE=-O3 -qopenmp -ipo -xCORE-AVX2 -ansi-alias -qopt-malloc-options=1
FOPTIMIZE=-fp-model fast=2 -no-prec-div -no-prec-sqrt -align array64byte

363.swim=peak=default=default:
OPTIMIZE=-O3 -qopenmp -ipo -xCORE-AVX2 -ansi-alias -qopt-malloc-options=4
threads=24
```

- `-qopt-malloc-options`: alternate algorithm for malloc
- `-fp-model fast=2`: aggressive optimization on FP computations
- `-no-prec-sqrt`: less precise square root computations/ more performance
- `-align array64byte`: align arrays to 64 Byte
- `-qopt-streaming-stores always`: use non-temporal stores (write through)
Peak runs

• Set of optimizations individually selected for each benchmark
  • e.g. different compilers, flags

• Called “aggressive compilation”

• Most published results do not contain peak, too much work

• lots of room to explore

• peak results are typically from vendors (e.g. showcase higher performance during RFP competition)

Peak optimization rules: https://www.spec.org/omp2012/Docs/runrules.html#section2.4
Published results OMP2012: https://www.spec.org/omp2012/results/omp2012.html
You will be able to play with the benchmarks in the hands-on section coming next.
I. SPEC benchmark philosophy
   a) Structure of SPEC benchmarks
      • Benchmark components & workloads
      • SPEC score
      • brief story of run rules
      • Configuration files
      • From base to peak runs
   b) Result Disclosure
      • Documentation and benchmark report
      • Reportable runs
      • the peer-review process for publishing

II. Use SPEC benchmarks for decision making
   • procurement process
   • test performance of compilers
   • compiler validation
   • explore performance of new architectures

III. Benchmark acquisition & licensing
   • commercial license
   • academic license
   • become a member
Reportable runs

Create valid/compliant result for submission:

```bash
runcspec --reportable [..]
```

- `--tune [base|all]`
- Entire SPEC suite (no single benchmarks)
- **Workload:** `test, train, ref` will be run → `ref` results are taken
  - Verification for all three data set sizes
- `#iterations = 3` → median is taken for final reporting
- `#threads`: one fixed number in base (variable per benchmark in peak)

Configuration disclosure (in config file or with `--rawformat` - see next slide)
Output Files

- SPEC runs automatically generates:
  - Benchmark reports: in txt, pdf, html formats
    - Preview of the result as it would look on the SPEC website
  - Raw files, "*.rsf",
    - Editable fields includes system and software configuration
    - Non-editable fields: benchmark results are encrypted. Tampering with the results will corrupt the file.
  - Log files, "*.log", "*.log.debug"
    - Verbose output of the benchmark run
Submitting results to SPEC

• Submission of SPEC results
  1. Process your rsf-file through rawformat to check for anything missing/ faulty
  2. Attach your rsf-file to an e-mail to, e.g., subaccel@spec.org for Accel
  3. Receive a reply with a sub-file attached
  4. For updates, modify the sub-file and attach to an e-mail to, e.g., resubaccel@spec.org for Accel

• Submitted results reviewed before publication by SPEC committee
  • Schedule: 2 weeks review period before decision

Source: https://www.spec.org/accel/results/accel_acc.html

OpenACC (21):

<table>
<thead>
<tr>
<th>Test Sponsor</th>
<th>System Name</th>
<th>Accelerator Name</th>
<th>Results Base</th>
<th>Energy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lenovo Global Technology</td>
<td>ThinkSystem SR650</td>
<td>NVIDIA Tesla V 100-PCIE-16GB</td>
<td>12.2</td>
<td>Not Run</td>
</tr>
<tr>
<td>Lenovo Global Technology</td>
<td>ThinkSystem SR670</td>
<td>Tesla V100-PCIE-16GB</td>
<td>12.0</td>
<td>Not Run</td>
</tr>
<tr>
<td>NVIDIA Corporation</td>
<td>SuperServer 1028GR-TR</td>
<td>Tesla K40m</td>
<td>2.56</td>
<td>2.56</td>
</tr>
<tr>
<td>NVIDIA Corporation</td>
<td>SuperServer 1028GR-TR</td>
<td>Tesla 100-PCIE-16GB</td>
<td>1.81</td>
<td>1.81</td>
</tr>
<tr>
<td>NVIDIA Corporation</td>
<td>SuperServer 1028GR-TR</td>
<td>Tesla P100-PCIE-16GB</td>
<td>8.02</td>
<td>8.02</td>
</tr>
</tbody>
</table>

Source: https://www.spec.org/accel/docs/runrules.html
https://www.spec.org/hpg/submitting_results.html
All SPEC Official results are publicly available on SPEC website: https://www.spec.org/results.html

Over 1400 published results, rich database for performance study.
full fair use policy: https://www.spec.org/fairuse.html

SPEC can go after any misuse (under license agreement).
A few recent cases:

- (osgmail-54973) Mar-2020, CPU maker A found CPU maker B claimed “36% more performance” than previous gen product using SPEC benchmarks.
  - no basis of comparison or how the speedup was calculated
  - no SPEC official metric was used
  - unpublished SPEC results need to be marked as “estimate”
  - company B eventually changed their press release with SPEC suggested changes.

- (osgmail-54450) Dec-2019. It’s found production machine from vendor X has different software/hardware setup, the performance was not reproducible. Their 12 published results were marked non-compliant with score remoted.

- (osgmail-51786) Jan-2019. Company C who was a relatively new member released new chip claiming certain SPEC score without submitting report to SPEC for review.
  - Company C was made to modify their press release with clear mark that the score was unofficial estimate.

- (osgmail-50759) Jun-2018. Vendor Y acquired machine made by vendor Z, and found they cannot achieve the same performance as their published results. 4 results were sent back for re-review.

……...
SPEC encourages use of its benchmarks in research and academic contexts. However, research use of SPEC benchmarks may not be able to meet the compliance requirement.

When making your result public (conference, press release, etc.), the requirements for Fair Use in academic/research contexts are:

1. It is a Fair Use violation to imply, to the reasonable reader, that a non-compliant number is a compliant result.
2. Non-compliance must be clearly disclosed. Mark the result as “estimate” or “experimental”, and explain the non-compliance, e.g. source code changes, experimental hardware/software, etc.
3. Diagrams, Tables, and Abstracts must have sufficient context on their own so that they are not misleading as to compliance.
4. If non-compliant numbers are compared to compliant results it must be clear from the context which is which.

https://www.spec.org/fairuse.html#Academic
Summary of SPEC Benchmark Philosophy

- SPEC benchmarks includes real applications
- SPEC tools maintenances consistency of the runs and check correctness
- SPEC score is speedup comparing to a reference machine, higher is better
- Each benchmark has its run rule
- All performance related aspects need to be documented in the report for reproducibility
- All results submitted to SPEC for publication will go through 2-week peer-review
- Feel free to use SPEC benchmarks for your research, just mark your result as “estimate” if it is not compliant with run rule
I. SPEC benchmark philosophy

   a) Structure of SPEC benchmarks
      • components & workloads & tools
      • SPEC score
      • brief story of run rules
      • Configuration files
      • From base to peak runs

   b) Result Disclosure
      • Documentation and benchmark report
      • Reportable runs
      • the peer-review process for publishing

II. Use SPEC benchmarks for decision making

   § procurement process
   § test performance of compilers
   § compiler validation
   § explore performance of new architectures

III. Benchmark acquisition & licensing

   § commercial license
   § academic license
   § become a member
Use SPEC benchmarks for decision making:
procurement process

RFP
- all major HPC vendors are SPEC members
- they are all familiar with SPEC benchmarks
- easier for vendors to provide you SPEC results than testing your customized application
- application benchmark gives a better correlation to the real life performance
- SPEC peer-review process guarantees fairness

Acceptance Test
- make sure real applications work well
- compare performance with published SPEC results
- test and find optimal software stack (default compiler, default MPI, etc.)
Use SPEC benchmarks for decision making:

test performance of compilers

**Compiler Performance on OMP2012**

- no improvement for GNU
- consistent high performance for Intel (especially for Fortran)
- descent improvement over 5 years for PGI (major improvement in Fortran)

**OMP2012 Score (Estimate): Dual Xeon E5-2650 v3**
(higher is better)
Use SPEC benchmarks for decision making:

test performance of compilers

**Compiler Validation**
- GNU OpenACC support under development
- SPEC Accel OpenACC benchmark being used by IU, ORNL, GNU/Mentor Graphics

### SPEC Estimated Scores on V100

<table>
<thead>
<tr>
<th>benchmark name</th>
<th>GCC/8.1.1 ACC branch *</th>
<th>PGI/18.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>303.ostencil</td>
<td>2.38</td>
<td>13.3</td>
</tr>
<tr>
<td><strong>304.olbm</strong></td>
<td><strong>13.6</strong></td>
<td><strong>11.3</strong></td>
</tr>
<tr>
<td>314.omriq</td>
<td>2.76</td>
<td>19.0</td>
</tr>
<tr>
<td>360.ilbdc</td>
<td>7.55</td>
<td>10.1</td>
</tr>
<tr>
<td><strong>Total Score</strong></td>
<td><strong>0.40</strong></td>
<td><strong>9.96</strong></td>
</tr>
</tbody>
</table>

(All codes)

- With some hacks, GCC compiles all 15 benchmarks
- GCC currently will run "acc kernels" serially on CPU.
- GCC handles "acc parallel" well.
- 4 out of 15 ACC benchmarks do not use "acc kernels"

* Info about GCC OpenACC branch: [https://gcc.gnu.org/wiki/OpenACC](https://gcc.gnu.org/wiki/OpenACC)
Use SPEC benchmarks for decision making:
Compare performance of different architectures

GPU vs KNL, which worth investment?
- SPEC Accel OpenACC and OpenMP have same workload, same serial code, different directives
- based on runtime from in-house test data, published SPEC data and estimate from core/freq

### OpenMP 4.5 (Intel 17)
- 6.4 x estimate
- 5.6 x
- 4.7 x
- 4.6 x
- 3.1 x
- 2.4 x
- 1.4 x
- 1.3 x

### OpenACC (PGI 17)
- 12.4 x Tesla P100
- 3.9 x Tesla K40m
- 3.4 x Tesla K20x
- 2.6 x
- 1.8 x
- 1.0 x *

* Use dual Xeon E5-2640 v2 OpenACC as baseline

derived speedup based on runtime in SPEC Accel OMP and SPEC Accel ACC

SC20 SPEC Tutorial, Part C: How to use SPEC benchmarks for decision making
I. SPEC benchmark philosophy

a) Structure of SPEC benchmarks
   • Benchmark components & workloads
   • SPEC score
   • brief story of run rules
   • Configuration files
   • From base to peak runs

b) Result Disclosure
   • Documentation and benchmark report
   • Reportable runs
   • the peer-review process for publishing

II. Use SPEC benchmarks for decision making

   - procurement process
   - test performance of compilers
   - compiler validation
   - explore performance of new architectures

III. Benchmark acquisition & licensing

   - commercial license
   - academic license
   - become a member
Acquisition of SPEC Benchmarks

- Single SPEC suites
- Commercial license
- Non-profit license
- SPEC membership
- Receive benchmarks for free

http://spec.org/order.html
https://www.spec.org/hpg/joining.html
Acquisition of SPEC Benchmarks

• Commercial license
  • Must be purchased via order form
  • Commercial enterprises (not being SPEC/HPG member) engaging in marketing, developing, testing, consulting for and/or selling computers, computer services, accelerator devices, drivers, software or other high performance computing systems or components in the computer marketplace

• Non-commercial license
  • Free of charge
  • Organizations that do not require a commercial license
  • Valid for the organization (not individual)
  • Institutional e-mail address required

Benchmark Suite | Non-Profit | Commercial
---|---|---
CPU2017 V1.0.2 | $250 | $1,000
ACCEL V1.3 | free | $2,000
OMP2012 V1.0 | free | $2,000
MPI2007 V2.0.1 | free | $2,000
SPECpower_ssj2008 V1.12 | $400 | $1,600

Non-commercial download + definition: http://spec.org/hpgdownload.html

SC20 SPEC Tutorial, Part C: How to use SPEC benchmarks for decision making
Free Non-commercial download + definition: http://spec.org/hpgdownload.html

- Get your free non-commercial license
- research labs, universities, government org, etc.
- as simple as filling out a lightweight form
- provide your institutional email for verification
- SPEC office will check your eligibility and send you the download link

- As of Q3-2020, 270 institutions have requested and been granted free license.
  (multiple requests from the same institute counts as one)
Become a member

- get access to all the benchmarks for free
- connect with all chip makers, vendors, compiler developers, universities, national labs, etc.
- learn the last development in software and hardware
- join the team work creating future benchmarks

- commercial member: $8000/y (more voting rights)
- academic member: $800/y

https://www.spec.org/spec/membership.html

please inquiry the SPEC office: info@spec.org
Thank you!

Questions?

Contact
SPEC Headquarters:
Robert Henschel:
Verónica G. Vergara Larrea:
Junjie Li:
Sunita Chandrasekaran:

info@spec.org
henschel@iu.edu
vergaravg@ornl.gov
lijunj@iu.edu
schandra@udel.edu

SC20 SPEC Tutorial, Part C: How to use SPEC benchmarks for decision making