A Retrospective Look at SPEC Benchmarking

Including: lessons learned

Prepared for: ModSim 2022
Workshop on Modeling & Simulation of Systems and Applications
Hosted by Brookhaven National Laboratory

John L. Henning
Performance Engineer, Oracle Corporation
Secretary, SPEC CPU Subcommittee

August 2022

A Retrospective Look at SPEC Benchmarking

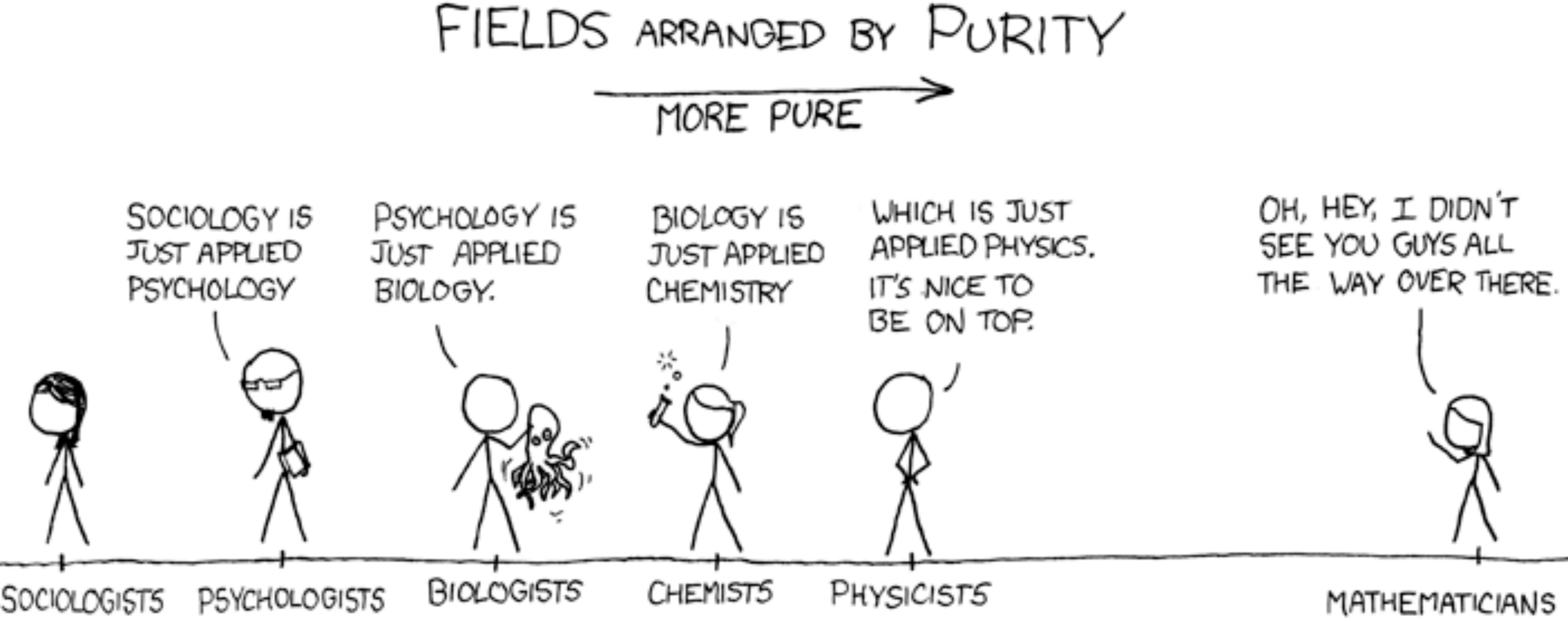
Including: lessons learned

Wait, what?
The modeling people invited a measurement guy?

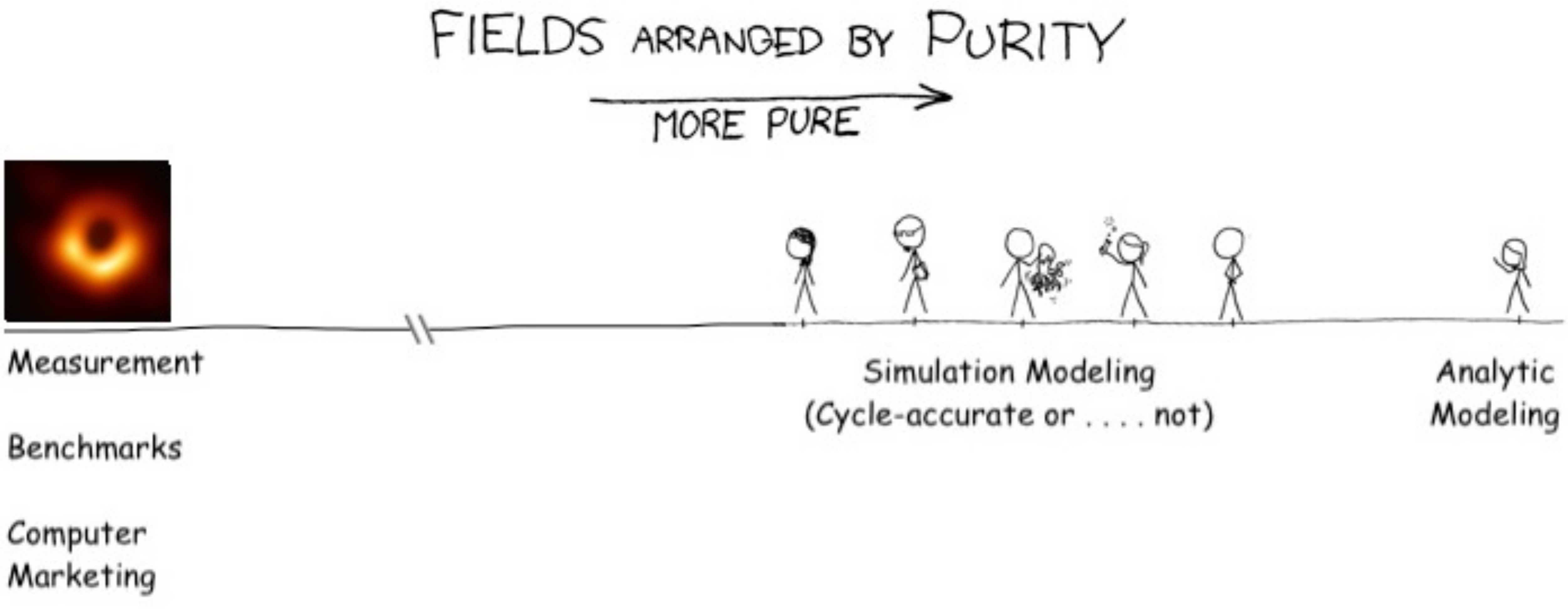
Prepared for: ModSim 2022
Workshop on Modeling & Simulation of Systems and Applications
Hosted by Brookhaven National Laboratory

John L. Henning
Performance Engineer, Oracle Corporation
Secretary, SPEC CPU Subcommittee

August 2022


FIELDS ARRANGED BY PURITY
MORE PURE
SOCIOLOGY IS JUST APPLIED PSYCHOLOGY
PSYCHOLOGY IS JUST APPLIED BIOLOGY.
BIOLOGY IS JUST APPLIED CHEMISTRY
WHICH IS JUST APPLIED PHYSICS. IT'S NICE TO BE ON TOP.
OH, HEY, I DIDN'T SEE YOU GUYS ALL THE WAY OVER THERE.
SOCIOLOGISTS
PSYCHOLOGISTS
BIOLOGISTS
CHEMISTS
PHYSICISTS
MATHEMATICIANS

xkcd #435 [1]


FIELDS ARRANGED BY PURITY
MORE PURE
Measurement
Benchmarks
Computer
Marketing
Simulation Modeling
(Cycle-accurate or . . . . not)
Analytic
Modeling

Why SPEC?

- Goal: an ounce of honest data is worth a pound of marketing hype.[2]

- Even if 1970s and 1980s benchmarkers are honest, **comparable** measurements are a lot harder than measurements.

- Case study: a 1985 performance guide. [3] [4]
  - 164 glossy, typeset pages
  - Allegedly “Internal Use Only”

Instruction timing for 7 different CPU models

10 pages of detailed comparisons

Table 4-2 • Instruction Timings Affected by FPA: FPA ON
High-End Systems Relative to Previous Model

Instruction		Raw Speed	Speed Relative to Previous Model	Raw Speed	Speed Relative to Previous Model	Raw Speed	Speed Relative to Previous Model	Ra
F_floating								
ADDF2	Reg, Reg	0.160	4.906	0.522	1.504	0.785	1.0	
ADDF3	Reg, Reg, Reg no alignment shift	0.241	5.091	0.786	1.561	1.227	1.0	
MULF2	Reg, Reg	0.321	3.763	0.803	1.504	1.208	1.0	
SUBF3	Reg, Reg, Reg alignment shift = 9	0.241	5.012	0.788	1.533	1.208	1.0	
DIVF3	Reg, Reg, Reg	1.367	3.388	3.065	1.511	4.631	1.0	
POLYF	(4TH ORDER)	3.451	2.755	6.306	1.508	9.508	1.0	
D_floating								
ADDD2	Reg, Reg	0.402	3.505	0.937	1.504	1.409	1.0	
ADDD3	Reg, Reg, Reg no alignment shift	0.564	4.284	1.459	1.656	2.416	1.0	
MULD2	Reg, Reg	1.045	3.263	2.270	1.502	3.410	1.0	
SUBD3	Reg, Reg, Reg alignment shift = 9	0.563	4.648	1.593	1.643	2.617	1.0	
DIVD3	Reg, Reg, Reg	5.309	1.669	5.878	1.507	8.861	1.	
POLYD	(4TH ORDER)	6.590	2.834	11.529	1.506	17.360	1	

Instruction timing for 7 different CPU models

10 pages of detailed comparisons

But . . .

- Only within that vendor's systems.
- The benchmark is not public.
- The benchmark is in assembler.

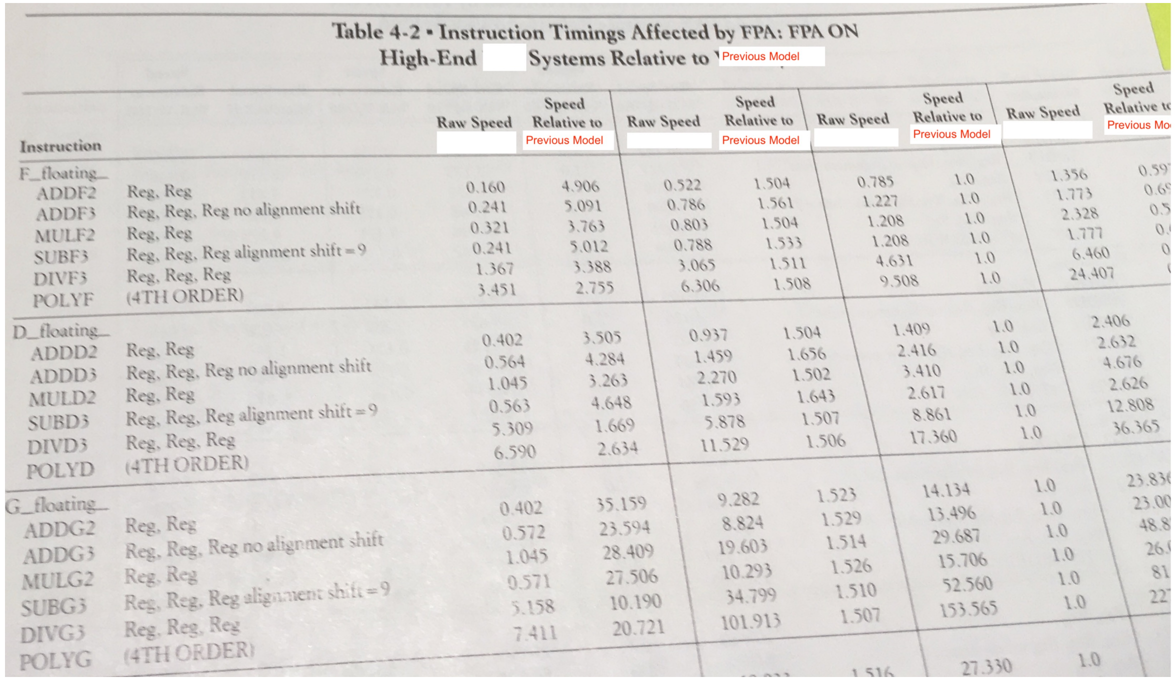


Table 4-2 • Instruction Timings Affected by FPA: FPA ON
High-End Systems Relative to Previous Model

Instruction		Raw Speed	Speed Relative to Previous Model	Raw Speed	Speed Relative to Previous Model	Raw Speed	Speed Relative to Previous Model	Raw Speed	Speed Relative to Previous Model
F_floating									
ADDF2	Reg, Reg	0.160	4.906	0.522	1.504	0.785	1.0	1.356	[illegible]
ADDF3	Reg, Reg, Reg no alignment shift	0.241	5.091	0.786	1.561	1.227	1.0	1.773	[illegible]
MULF2	Reg, Reg	0.321	3.763	0.803	1.504	1.208	1.0	2.328	[illegible]
SUBF3	Reg, Reg, Reg alignment shift=9	0.241	5.012	0.788	1.533	1.208	1.0	1.777	[illegible]
DIVF3	Reg, Reg, Reg	1.367	3.388	3.065	1.511	4.631	1.0	6.460	[illegible]
POLYF	(4TH ORDER)	3.451	2.755	6.306	1.508	9.508	1.0	24.407	[illegible]
D_floating									
ADDD2	Reg, Reg	0.402	3.505	0.937	1.504	1.409	1.0	2.406	
ADDD3	Reg, Reg, Reg no alignment shift	0.564	4.284	1.459	1.656	2.416	1.0	2.632	
MULD2	Reg, Reg	1.045	3.263	2.270	1.502	3.410	1.0	4.676	
SUBD3	Reg, Reg, Reg alignment shift=9	0.563	4.648	1.593	1.643	2.617	1.0	2.626	
DIVD3	Reg, Reg, Reg	5.309	1.669	5.878	1.507	8.861	1.0	12.808	
POLYD	(4TH ORDER)	6.590	2.634	11.529	1.506	17.360	1.0	36.365	
G_floating									
ADDG2	Reg, Reg	0.402	35.159	9.282	1.523	14.134	1.0	[illegible]	
ADDG3	Reg, Reg, Reg no alignment shift	0.572	23.594	8.824	1.529	13.496	1.0	[illegible]	
MULG2	Reg, Reg	1.045	28.409	19.603	1.514	29.687	1.0	[illegible]	
SUBG3	Reg, Reg, Reg alignment shift=9	0.571	27.506	10.293	1.526	15.706	1.0	[illegible]	
DIVG3	Reg, Reg, Reg	5.158	10.190	34.799	1.510	52.560	1.0	[illegible]	
POLYG	(4TH ORDER)	7.411	20.721	101.913	1.507	153.565	1.0	[illegible]	
				[illegible]	1.516	27.330	1.0		

- Customers do not know whether their applications use fast or slow instructions.

“Industry Standard” Applications

Excerpts [no ellipses markings]

- industry-standard FORTRAN benchmarks are described in Table 6-13. Several have been altered to reduce variability.
- Harris Test is an industry-accepted benchmark that exercises features commonly found in APL programs. Many vendors have advertised results. Because data arrays are often limited in these advertisements, be aware that they may not accurately represent performance.
- IFTEST and LITTL have zero elapsed time. FORTRAN version 4 detects “dead code” yielding zero elapsed runtime.

“Industry Standard” Applications

Not in assembly, and claimed to be widely available.

But …

- Unclear which versions were used.
- Source code was changed.   The exact changes are unclear.
- Dead code is present.

Transaction processing

- 40 pages of detail about orders, inventory, sales, receiving.
- Enormous effort building RTE (remote terminal emulators), workload, systems under test.

But . . .

- Compares only a single vendor’s systems.
- Benchmark sources are not available.

Whetstone and Dhrystone

- References were provided to specific source versions.
- Dhrystone includes “run rules”, reducing ambiguity.

Whetstone and Dhrystone

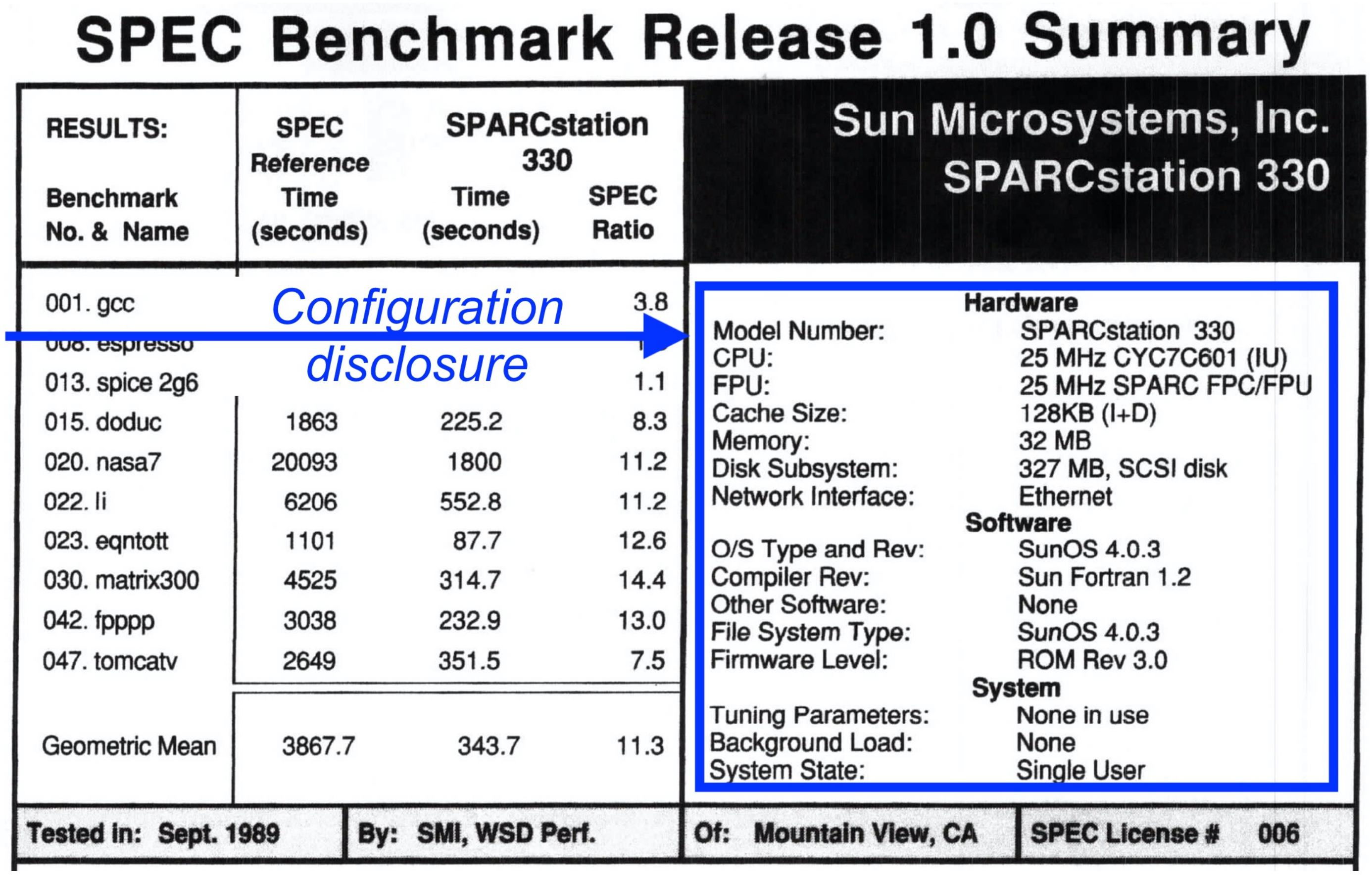
- References were provided to specific source versions.
- Dhrystone includes “run rules”, reducing ambiguity.

But . . .

- Both had several versions, and advertisements were not always clear about which was used.
- Both were “synthetic” collections of program fragments.
- Dhrystone did not have a rule enforcement mechanism.

SPECmark – lessons learned vs. predecessors

- Start from real application programs, which are more meaningful than instruction times and synthetic kernels
- In order to enable comparisons:
  - Require full disclosure of test conditions



SPEC Benchmark Release 1.0 Summary

Sun Microsystems, Inc.
SPARCstation 330

RESULTS: Benchmark No. & Name	SPEC Reference Time (seconds)	SPARCstation 330 Time (seconds)	SPEC Ratio
001. gcc			3.8
008. espresso			
013. spice 2g6			1.1
015. doduc	1863	225.2	8.3
020. nasa7	20093	1800	11.2
022. li	6206	552.8	11.2
023. eqntott	1101	87.7	12.6
030. matrix300	4525	314.7	14.4
042. fpppp	3038	232.9	13.0
047. tomcatv	2649	351.5	7.5
Geometric Mean	3867.7	343.7	11.3

Hardware	
Model Number:	SPARCstation 330
CPU:	25 MHz CYC7C601 (IU)
FPU:	25 MHz SPARC FPC/FPU
Cache Size:	128KB (I+D)
Memory:	32 MB
Disk Subsystem:	327 MB, SCSI disk
Network Interface:	Ethernet
Software	
O/S Type and Rev:	SunOS 4.0.3
Compiler Rev:	Sun Fortran 1.2
Other Software:	None
File System Type:	SunOS 4.0.3
Firmware Level:	ROM Rev 3.0
System	
Tuning Parameters:	None in use
Background Load:	None
System State:	Single User

Tested in: Sept. 1989	By: SMI, WSD Perf.	Of: Mountain View, CA	SPEC License # 006

SPECmark – lessons learned vs. predecessors

- Start from real application programs, which are more meaningful than instruction times and synthetic kernels
- In order to enable comparisons:
  - Require full disclosure of test conditions
  - Require that everyone test the same source code and data.
  - Specify run rules.
  - Review each others' rule compliance.
- Validate program output

SPECmark – lessons learned vs. predecessors

- Start from real application programs, which are more meaningful than instruction times and synthetic kernels
- In order to enable comparisons:
  - Require full disclosure of test conditions
  - Require that everyone test the same source code and data.
  - Specify run rules.
  - Review each others' rule compliance.
- Validate program output

"I can make it run as fast as you like if you remove the constraint of getting correct answers."

SPEC met with rapid success

- By 1995, result publications included: Bull, CDC, Compaq, Cray, Dansk, Data General, DEC, Gateway, Hitachi, HAL, HP, IBM, Intel, Intergraph, Micronics, Motorola, Pyramid, SGI, Siemens-Nixdorf, Solbourne, Sun, Tricord, Unisys [5]

- ISCA '95: Proceedings Of The 22nd Annual International Symposium On Computer Architecture: 16 papers used SPEC benchmarks [6]

Not to mention that it was fun!

Public domain photo by Emiliano Russo from the Wikipedia VAX-11 page

VAX 11/780
About 1100 lb (500 kg)

SPARCstation
About 10 lb
(not incl. monitor)

Creative Commons photo by Thomas Kaiser from the Wikipedia SPARCstation 10 page

SPECmark showed that 1989 single-chip systems were ~10x as fast as the 1978 VAX 11/780

SPEC Benchmark R

RESULTS: Benchmark No. & Name	SPEC Reference Time (seconds)	SPARCstation 330 Time (seconds)	SPARCstation 330 SPEC Ratio
001. gcc	1482	107.6	13.8
008. espresso	2266	195.9	11.6
013. spice 2g6	23951	2152.6	11.1
015. doduc	1863	225.2	8.3
020. nasa7	20093	1800	11.2
022. li	6206	552.8	11.2
023. eqntott	1101	87.7	12.6
030. matrix300	4525	314.7	14.4
042. fpppp	3038	232.9	13.0
047. tomcatv	2649	351.5	7.5
Geometric Mean	3867.7	343.7	11.3

Number of Published Results – SPEC CPU [7]

958	SPEC CPU 92
2,574	SPEC CPU 95
7,654	SPEC CPU 2000
48,381	SPEC CPU 2006
28,357	SPEC CPU 2017 as of Aug-2022

SPEC also published benchmark results for:

file servers | web servers
virtualization | mail servers

```
142 SPEC SFS 93
506 SPEC SFS 97
120 SPEC SFS 2008
 52 SPEC SFS 2014
 29 SPECstorage Solution 2020

 35 SPECvirt_sc 2010
 59 SPECvirt_sc 2013
  1 SPECvirt Datacenter 2021
```

```
231 SPECweb 96
371 SPECweb 99
100 SPECweb 2005
  7 SPECweb 2009

 18 SPECmail2001
  2 SPECmail2008
  5 SPECmail2009
```

SPEC also published Java benchmark results

```
 96 SPECjvm 98

 11 SPECjvm 2008
  4 SPECjms 2007

366 SPECjbb 2000
761 SPECjbb 2005
 90 SPECjbb 2013
695 SPECjbb 2015
```

```
14 SPECjAppServer  2001
37 SPECjAppServer  2002
95 SPECjAppServer  2004
50 SPECjEnterprise 2010
```

SPEC also published benchmarks for:

Software Development Environment
High Performance Computing
Power

```
 30 SPEC SDM 94

 60 SPEC HPC 96                 175 SPEC OMP 2012
104 SPEC HPC 2002               617 SPEC MPI 2007
 87 SPEC HPC 2021               135 SPEC ACCEL 2014

827 SPECpower_ssj2008
```

Wildly ambiguous

Thus for over 25 years, it has been wildly ambiguous to refer to:

- “SPEC results”
- “SPEC95”
- “SPEC2000”

The correct form is: SPEC <benchmark> <year>

Wildly ambiguous

Thus for over 25 years, it has been wildly ambiguous to refer to:

- “SPEC results”
- “SPEC95”
- “SPEC2000”

The correct form is: SPEC <benchmark> <year>

Lesson learned: nobody forgets your childhood nickname.

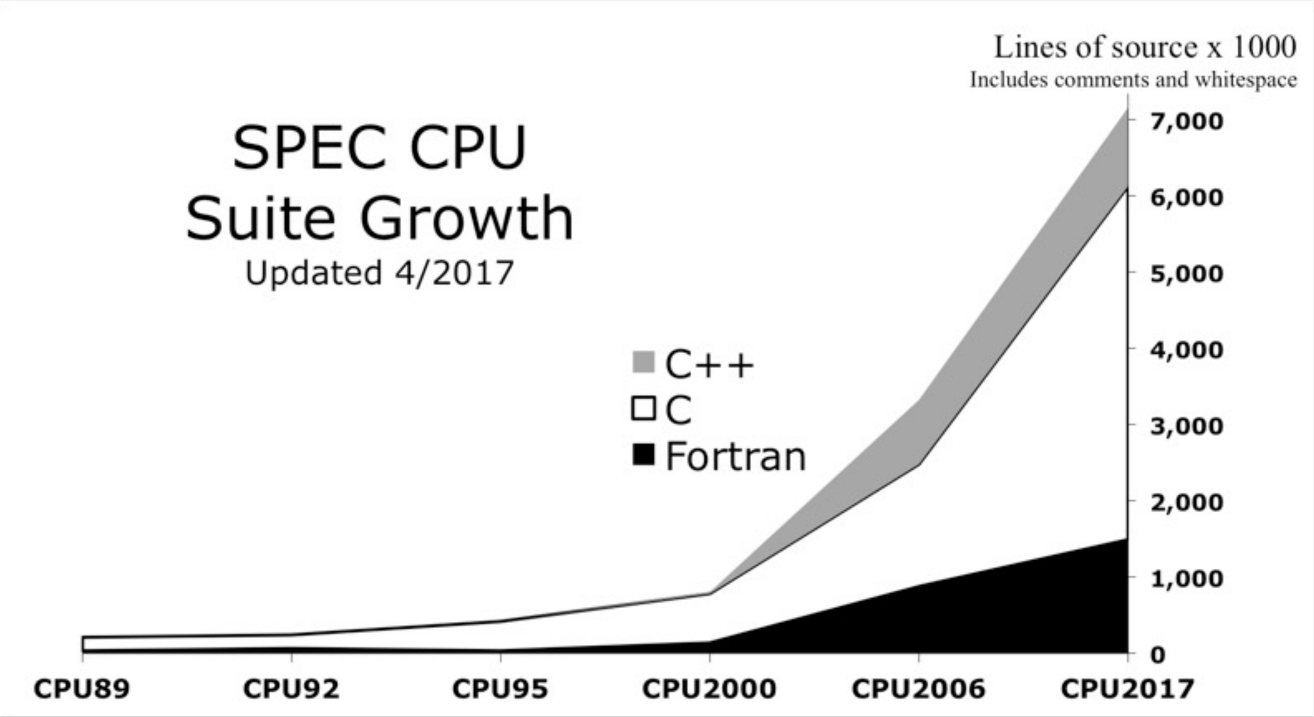
SPEC CPU Further Evolution / Lessons Learned

Prefer real apps. SPECmark89 started this; it expanded with each subsequent suite.

Read input data.
SPECmark89 had some programs where the compiler could see everything.

Centralize all user changes to 1 file.
In 1989, there were many Makefiles

Full disclosure is good. Fuller is better.
Requirements were added to disclose more detail, and several kinds of automatic disclosure were added. [8]



Lessons specifically from SPEC CPU 2017

- As noted by Professor Lizy John[9], there was an 11-year gap between CPU 2006 and SPEC CPU 2017.
- What took so long?

Wait of a Decade: Did SPEC CPU 2017 Broaden the Performance Horizon?

Reena Panda†, Shuang Song†, Joseph Dean, Lizy K. John
The University of Texas at Austin
reena.panda@utexas.edu, songshuang1990@utexas.edu, jd45664@utexas.edu, ljohn@ece.utexas.edu

Lessons learned from SPEC CPU 2017 (1)

Pick parallelization targets early

- SPEC CPU 2017 evaluated benchmark candidates that used:

Unix pthreads	Windows threads
OpenMP	MPI
Intel Threading Building Blocks	

These led to continual arguments, such as:

"We can easily build a shim layer."	"It's not so easy"
"System X can't run benchmark Y"	"Who cares?"
"Do we <u>require</u> parallelism?	"For which benchmarks?"

Lessons learned from SPEC CPU 2017 (2)

(re-learned) Kernels are less interesting than real apps

- One author submitted 11 small candidates
- None survived the final cut

Lessons learned from SPEC CPU 2017 (3)

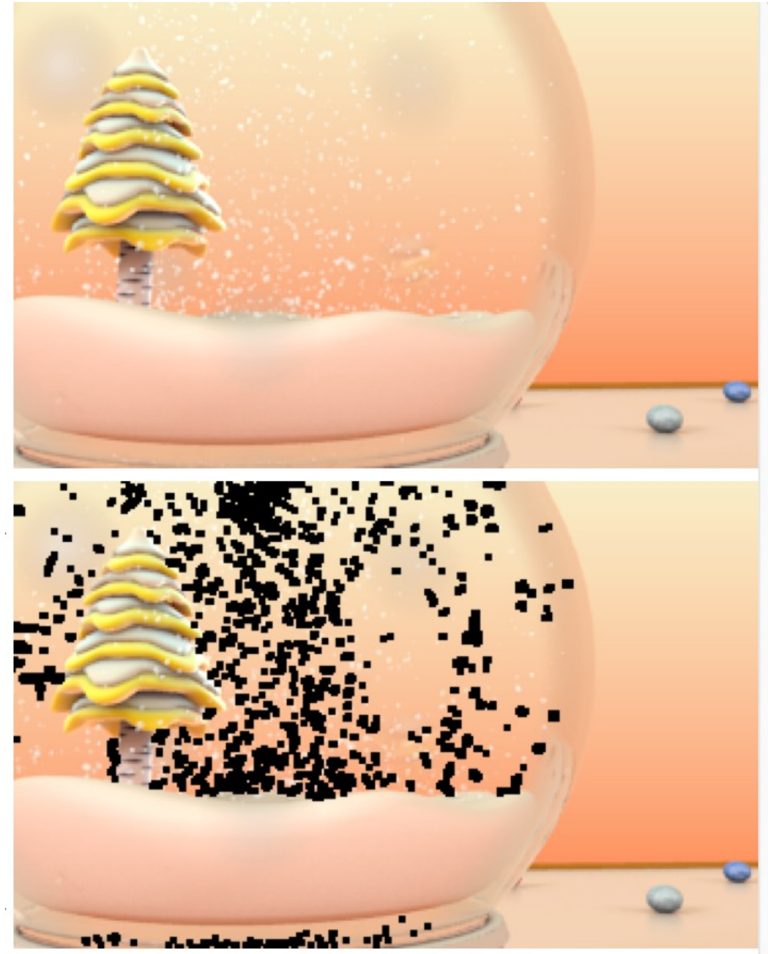
Specify resources (time / memory) up front

- There were too many arguments about how-long-is-too-long.
- And how-big-is-too-big.
- And how these are to be measured.

Lessons learned from SPEC CPU 2017 (4)

Figure out validation early

- Multiple benchmark candidates had no validation method.
- SPEC's image validator[10] implements SSIM, which solved the problem for several benchmarks.



Lessons learned from SPEC CPU 2017 (5)

(re-learned) Portability is hard

- One author submitted 17 candidates claiming that all were highly portable.
- The basis for this claim turned out to be that the benchmarks had indeed been tested on multiple hardware types, but:
  - using only one compiler
  - on one operating system

Lessons learned from SPEC CPU 2017 (6)

(re-learned) Licenses are important

- Each version of SPEC CPU has drawn on the open source community, starting with SPECmark89 and 001.gcc from the Free Software Foundation
- A license review near the end of the SPEC CPU 2017 development cycle turned up a few surprises, such as:

```
regexp.c
Copyright (c) 1984 AT&T All Rights Reserved THIS IS
UNPUBLISHED PROPRIETARY SOURCE CODE OF AT&T The copyright
notice above does not evidence any actual or intended
publication of such source code.
```

A different regexp solution was found. [11]

Lessons learned from SPEC CPU 2017 (7)

Recurring benchmarking pitfalls will recur

In addition to the various benchmark candidate pitfalls mentioned above, some other recurring problems include:

- A benchmark candidate measures something different than expected.
- The benchmark candidate is a library with hundreds of functions. The workload exercises 2 of them.
- Minor FP differences unexpectedly cause differing work.

A list of good[12] and bad[13] benchmark characteristics was published with the release of SPEC CPU 2017.

Lessons learned from SPEC CPU 2017 (8)

Subjective criteria are inevitable; don’t wait until the end to expose them

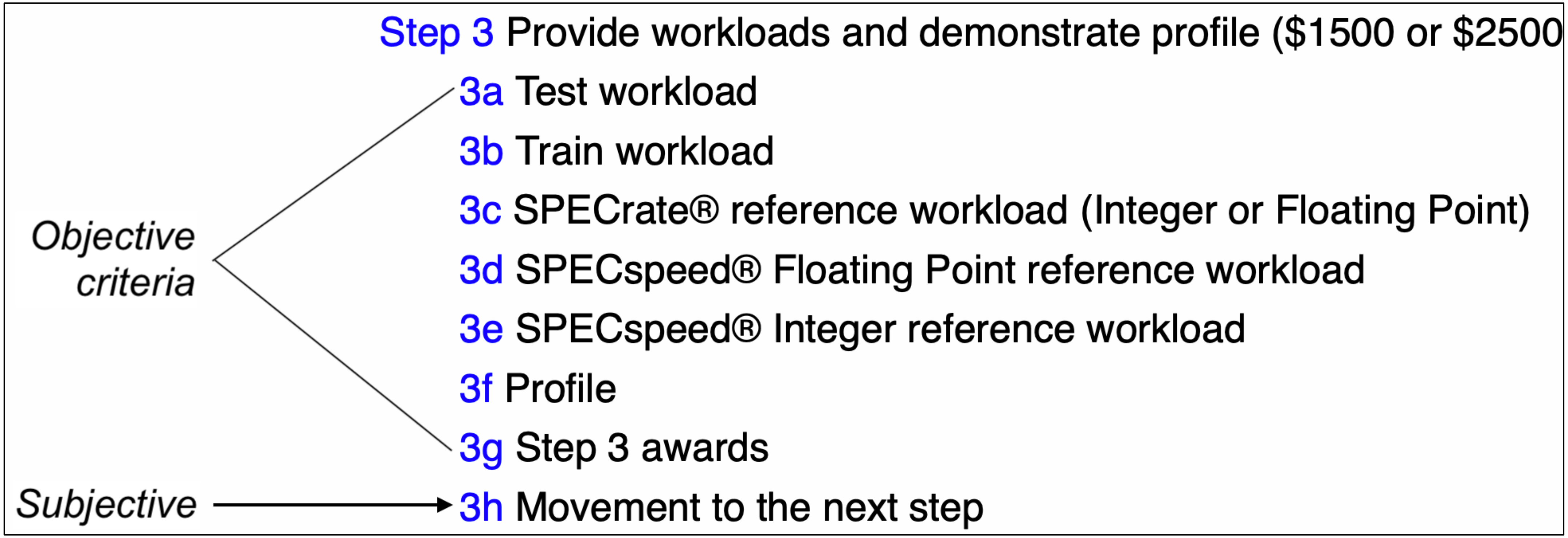
What is most important among these?

Application domain	Modern code
Large user base	Exercises main memory
Well-known application	Non-peaky profile
State of the art algorithm	Expert author
Not ”just a toy” program	Avoids repeating identical work
Robust code	Easily ported to new systems
Compiler challenge	Not previously in a SPEC CPU suite

Applied lessons - SPEC CPUv8 Search program[14]

www.spec.org/cpuv8

- Requires both objective and subjective criteria at each step



Applied lessons - SPEC CPUv8 Search program

www.spec.org/cpuv8

- Requires both objective and subjective criteria at each step
- Specifies parallelization technologies
- Specifies which suites require parallelism.
- Specifies run time, memory size, and how these are measured.
- Requires early review of licenses

Speaking of the SPEC CPUv8 Search program

www.spec.org/cpuv8

- It’s not too late
- Fill out your entry form by 12-Sep-2022
- (You do not have to finish all the steps by 12-Sep! Just get the process started.)

References

[1] R. Munroe, “Purity”, https://xkcd.com/435/

[2] Standard Performance Evaluation Corporation, “SPEC Organizational Information”, https://www.spec.org/spec/ section “Background”.

[3] The material in the next several slides is based on J. L. Henning, "How Many VAXes Fit in the Palms of Your Hands?," in IEEE Micro, vol. 41, no. 6, pp. 140-143, 1 Nov.-Dec. 2021, doi: 10.1109/MM.2021.3112911. The author’s pre-production copy is available at https://www.spec.org/cpu2017/publications/HowManyVAXesFitInThePalmsOfYourHands-AuthorVersion.pdf

[4] As explained at [3], a full citation for the 1985 Performance Summary is not provided because it is marked as internal use only. In practice, considering the amount of effort that went into its production, it seems likely that some customers were given copies.

[5] The list of vendors is based on SPEC results from the early 1990s as archived at https://netlib.org/performance/html/spec.html

[6] Association for Computing Machinery, Proceedings of the 22nd annual international symposium on Computer architecture, 1995, New York, NY, USA. Free access at https://dl.acm.org/doi/proceedings/10.1145/223982

[7] The result counts are based the totals reported for each benchmark as linked from Standard Performance Evaluation Corporation, “Published SPEC Benchmark Results”, https://www.spec.org/results.html and ”Retired Benchmarks”, https://www.spec.org/retired.html. The CPU92 counts are from counting the archived pages at [5].

[8] See for example Standard Performance Evaluation Corporation, “SPEC CPU2017 Run and Reporting Rules”, https://www.spec.org/cpu2017/Docs/runrules.html, section 4.

[9] R. Panda, S. Song, J. Dean and L. K. John, "Wait of a Decade: Did SPEC CPU 2017 Broaden the Performance Horizon?," 2018 IEEE International Symposium on High Performance Computer Architecture (HPCA), 2018, pp. 271-282, doi: 10.1109/HPCA.2018.00032.

[10] Standard Performance Evaluation Corporation, “SPEC CPU 2017 Utilities”, https://www.spec.org/cpu2017/Docs/utility.html section “imagevalidate”

[11] Standard Performance Evaluation Corporation, “SPEC CPU 2017 Licenses”, https://www.spec.org/cpu2017/Docs/licenses.html

[12] Standard Performance Evaluation Corporation, “SPEC CPU 2017 Overview / What's New?”, https://www.spec.org/cpu2017/Docs/overview.html

[13] Ibid, section Q3.

[14] Standard Performance Evaluation Corporation, “SPEC CPU v8 Benchmark Search Program”, https://www.spec.org/cpuv8/