



---

# Lessons from SPEC

John R. Mashey (mash) AT techviser dot com  
ACL-2021 Workshop on Benchmarking:  
Past, Present and Future (BPPF)  
August 5-6, 2021

# Speaker – John R. Mashey

---

- PhD Computer Science, Penn State, 1974
- Early UNIX software, Bell Labs 1973-1983, MTS → Supervisor
  - Programmer's Workbench, shell programming, text processing, environment variables, **workload measurement/tuning** in first UNIX computer center, **UNIX per-process accounting**
  - UNIX+mainframe data mining apps, **capacity planning/tuning**, **benchmarking** => 3B20
- Convergent Technologies 1983-1984, MTS → Director Software
  - Compiler & OS **tuning**, uniprocessor/multiprocessor servers
- MIPS Computer Systems 1985-1992, Mgr. OS → VP Systems Technology
  - System coprocessor, TLB, interrupt-handling; byte addressing(!), halfword instructions; ISA evolution, multiprocessor features, multi-page-size TLB, 64-bit
  - **MIPS Performance Brief editor; a SPEC benchmarking group founder 1988**
  - Hot Chips committee through 2015; ex-frequent poster on comp.arch
- Silicon Graphics 1992-2000, Director Systems Technology → Chief Scientist & VP
  - R10000 & later architecture, including **performance counters** & software, **Big Data**
  - ccNUMA system architecture (NUMAflex in Origin3000, Altix)   **1995**
  - **Benchmarking issues in High Performance Computing**, DBMS; technology forecasting
- Mostly-retired 😊 Occasional consulting for VCs / advisory committees for high-tech co's; Computer History Museum ([www.computerhistory.org](http://www.computerhistory.org)) Trustee since 2001  
2011-13 – consulting for Nvidia, Unified Memory, **performance**, ~17 patents
- Investigative reporting on climate denialism/disinformation, articles & talks since 2008  
Advisor UCSF Center for Tobacco Control Research & Education since 2013

# Overview

---

- 3+1 benchmarking organizations – TPC, SPEC, EEMBC, *MLCommons*
- SPEC in 2021
- Historical context for SPEC
- Origin of SPEC in Campbell bar and early days
- SPEC - Correct math, but missed statistical implications
- Lessons, including on brief thoughts on NLP benchmarking

“When you can measure what you are speaking of and express it in numbers, you know that on which you are discussing. But when you cannot measure it and express it in numbers, your knowledge is of a very meagre and unsatisfactory kind.” -Lord Kelvin

“...benchmarks shape a field, for better or worse. Good benchmarks are in alignment with real applications, but bad benchmarks are not, forcing engineers to choose between making changes that help end users or making changes that only help with marketing.” -David A. Patterson,

“Creating good benchmarks is harder than most imagine.” – John R. Mashey  
Forewords to *Systems Benchmarking (2020)*, by Kounev, Lange, Kistowski (SPEC)

No matter how much people want performance to be a single number, even the right mean with no distribution can be misleading, and the wrong mean certainly is no better.

# 3 Long-established benchmarking groups, 1 New

---

- Old ones
  - 1988 TPC – Transaction Processing Performance Council <http://www.tpc.org/>  
Market researcher Omri Serlin convinced 8 companies to form TPC, 08/10/88.
  - 1988 SPEC – Standard Performance Evaluation Corporation [www.spec.org](http://www.spec.org)  
*EE Times*' reply to email was impetus for 4 companies to start, incorporated 11/14/88.
  - 1997 EEMBC Embedded Microprocessor Benchmark Consortium [www.eembc.org](http://www.eembc.org)  
Originally started by *EDN* (Electrical Design News)' Markus Levy, as EDN EMBC, 12 members.
  - Long history may offer insights, but next, new group likely overlaps more with NLP:
- MLCommons <https://mlcommons.org> for Machine Learning, academe+industry
  - As per <https://rise.cs.berkeley.edu/blog/mlperf-spec-for-ml/>  
“The RISE Lab at UC Berkeley today joins Baidu, Google, Harvard University, and Stanford University...”  
“**Historical Inspiration.** We are motivated in part by the ... ([SPEC](http://www.spec.org)) benchmark for general-purpose computing that drove rapid, measurable performance improvements for decades starting in the 1980s.”
  - 2018 MLPerf first results <https://mlcommons.org/en/history/>
- All are driven, (mostly) funded by corporate members  
SPEC, EEMBC and MLCommons have significant participation by academics/labs.  
Groups sometimes overlap/cooperate, power metrics by SPEC and MLCommons:  
<https://www.hpcwire.com/2021/04/21/mlperf-issues-new-inferencing-results-adds-power-metrics-nvidia-wins-again/> Also EEMBC-MLCommons interactions

# SPEC - 2021

---

- SPEC - [www.spec.org](http://www.spec.org), [https://en.wikipedia.org/wiki/Standard\\_Performance\\_Evaluation\\_Corporation](https://en.wikipedia.org/wiki/Standard_Performance_Evaluation_Corporation)
  - “**The Standard Performance Evaluation Corporation (SPEC)** is a non-profit corporation formed to establish, maintain and endorse standardized benchmarks and tools to evaluate performance and energy efficiency for the newest generation of computing systems. SPEC develops benchmark suites and also reviews and publishes submitted results from our [member organizations](#) and other benchmark licensees.”
  - 2018 30 Years – meeting <https://www.spec.org/30th>  
Timeline <https://www.spec.org/30th/timeline.html> - 125 members in 22 countries
  - Most work done by members/associates, plus 4 staff
- SPEC started with 1 small group, now 5, often with multiple subgroups
  - 1988 OSG - Open Systems Group – “benchmarks for desktop systems, workstations & servers running open operating system environments” [www.spec.org/osg/](http://www.spec.org/osg/)
  - 1996 GWPG - Graphics & Workstation Performance Group, was GPC → [www.spec.org/gwpg](http://www.spec.org/gwpg)
  - 1996 HPG - High Performance Group (parallel supercomputing) [www.spec.org/hpg](http://www.spec.org/hpg)  
PERFECT Club → <https://www.spec.org/30th/hpg.html>
  - 2011 RG - Research Group – “collaborative research efforts in the area of quantitative system evaluation and analysis, fostering the interaction between industry and academia. It also researches and then suggests new groups/subgroups for developing areas.”
    - <https://research.spec.org/news/single-view/article/new-textbook-on-systems-benchmarking.html>  
Kounev, Lange, Kistowski, Systems Benchmarking (2020)
  - >2018 ISG - International Standards Group - work with governments [www.spec.org/isg/](http://www.spec.org/isg/)

# SPEC – Members and Associates

---

- <https://www.spec.org/consortium/>
- Each group (OSG, GWPG, HPG, ISG, RG) has
  - Own benchmark suites, rules and dues structure
  - Members – mostly companies, bulk of the money
  - Associates – mostly universities, sometimes National Laboratories, standards groups.
  - Some overlap, depending on interests
- OSG, for example.
  - Membership \$2,500 initiation, \$7,500 Annual Dues, get licenses for all benchmarks
  - Associate \$1,000 initiation \$ 500 Annual Dues (Nonprofits, educational)
- Companies typically devote 1 or more people to SPEC work, salaries far outweigh
- Benchmarks typically cost \$1500-\$2500 commercial license, noncommercial free.

# SPEC – OSG Members and Associates

---

- Example – OSG - <https://www.spec.org/consortium/>
- “Members:  
Advanced Micro Devices (AMD) \* Altos Computing Inc. \* Amazon Web Services, Inc. \* Apple Inc. \* ARM \* ASUSTek Computer Inc. \* Auristor Inc \* Bull Atos Technology \* Chengdu Haiguang IC Design Co., Ltd. \* Cisco Systems \* Dell Inc. \* Epsilon Sp. z.o.o. Sp. Komandytowa \* Format sp. z o.o. \* Fujitsu, Ltd. \* Gartner, Inc. \* Giga-Byte Technology Co., Ltd. \* Google, Inc. \* Hewlett Packard Enterprise (HPE) \* Hitachi Vantara LLC \* Hitachi, Ltd \* IBM \* Inspur \* Intel Corporation \* iXsystems Inc. \* Lenovo \* Microsoft Corporation \* NEC Corporation \* NetApp, Inc. \* Netrix \* Netweb Pte Ltd \* New H3C Technologies Co., Ltd. \* NVIDIA Corporation \* Oracle Corporation \* Principled Technologies, Inc. \* Pure Storage \* Qualcomm Technologies Inc. \* Quanta Computer Inc \* Red Hat, Inc \* Supermicro Computer, Inc. \* Taobao (China) Software Co. Ltd. \* VIA Technologies, Inc. \* VMware, Inc. \* WekaIO \* ZTE Corporation \*”
- “Associates:  
Charles University \* China Academy of Information and Communications Technology \* China Electronics Standardization Institute \* ForTISS -- An-Institut der Technischen Universitaet Muenchen \* Indiana University \* Institute of Information Science, Academia Sinica \* Japan Advanced Institute of Science and Technology (JAIST) \* Karlsruhe Institute of Technology (KIT) \* Leibniz-Rechenzentrum, Bavarian Academy of Science \* Linaro Limited \* National University of Singapore \* Peng Cheng Laboratory \* Purdue University \* RWTH Aachen University \* South China University of Technology \* Technische Universitat Darmstadt \* Technische Universitat Dresden, ZIH \* Telecommunications Technology Association \* Tsinghua University \* University of Aizu \* University of California at Berkeley \* University of L'Aquila - Dept of Computer Science and Engineering (DISIM) \* University of Maryland \* University of Miami \* University of Texas at Austin \* University of Tsukuba \* University of Wuerzburg \* UT-Battelle Oak Ridge National Labs (ORNL) \* Virginia Polytechnic Institute and State University \*”

# Benchmark categories

---

- Benchmarks are usually called SPEC <label> year, like SPECint92, SPECfp92, SPECviewperf 2020, etc.
- Benchmark sets - <https://www.spec.org/benchmarks.html> – multiple programs/set
  - (1) [Cloud](#) – OSG
  - (1) [CPU](#) - OSG (1989, 1992, 1995, 2000, 2006, 2017)- *how it started*
  - (12) [Graphics/Workstations](#) – GWPG
  - (3) [ACCEL/MPI/OMP](#) – HPG – (3)
  - (4) [Java Client/Server](#) - OSG
  - (2) [Storage](#) - OSG
  - (2) [Power](#) - OSG
  - (1) [Virtualization](#) – OSG
  - (30) Retired – benchmarks become obsolete, although some components persist
- Pervasive issue, not just for SPEC
  - People want to see individual benchmark results, select those most relevant
  - But also demand overall summary numbers, many historical arguments over math
  - Designers of hardware and software want to use summaries for optimization



# Historical context for SPEC

---

- 1970- Computer industry mainframes & minicomputers
  - Proprietary operating systems
  - Fortran and COBOL more-or-less portable, if careful
  - **System implementation languages proprietary to vendors, rarely released or supported outside**
  - Bell Labs' Unix, C language propagated widely around universities, starting ~1974 on Digital Equipment Corporation PDP-11. UC Berkeley active with Berkeley System Distribution (BSD)
  - Unix Edition VII (1979) relatively portable, included retargetable Portable C Compiler (PCC)
  - DEC VAX 11/780 (1978) most successful superminicomputer family, widely used, became canonical "1" in performance, ran both DEC VAX/VMS and Unix, esp. BSD
  - Motorola 68000 (1979) & 68010 (1982) adequate for running {Unix, C} → workstations, servers
- 1980- Rapid rise of microprocessors, especially Reduced Instruction Set Computers
  - Early 1980s – RISC research: UC Berkeley (David Patterson) and Stanford (John Hennessy), **strong emphasis on quantitative design methods based on benchmarks, not intuition**
  - ~1986- First "golden age of computer architecture", Hennessy & Patterson Turing lecture, paper <https://cacm.acm.org/magazines/2019/2/234352-a-new-golden-age-for-computer-architecture/fulltext>
  - "Cambrian explosion" of new microprocessor architectures  
Almost all had C compilers → for first time, common system implementation language
  - Rapid increases in performance, far beyond rates of superminis & mainframes
  - MIPS, Hewlett-Packard, IBM, others shipped global optimizing compilers  
→ sometimes caused trouble for simplistic benchmarks
  - **Ferocious competition, ever-changing industry coalitions, claims, arguments, technologies**

# Benchmarking and Benchmarking - 1988

---

- CPU performance seems most basic metric, one might think it easy, BUT...
- [https://en.wikipedia.org/wiki/Benchmark\\_\(computing\)](https://en.wikipedia.org/wiki/Benchmark_(computing)) reasonable overview  
Many date from the 1980s.
- Plethora of benchmarks, typically in C or Fortran, not very consistent
  - Academics produced many
  - *Unix Review*, *Digital Review*, *Byte*, etc published own sets of benchmarks
  - High-performance computing had (decent) floating point LINPACK, Livermore Loops, for (usually vector) codes, but small synthetic, unrealistic Whetstone widely quoted
  - Some of the most popular benchmarks were awful, especially **Dhrystone**, infamously unrealistic, small integer C program, susceptible to unrealistic optimizations, outright cheating
  - Customers wanted (often large) realistic benchmarks to which they could relate
  - **Customers/third-party developers brought programs to “porting centers” → reality check**
- Most computer companies regularly published “Performance Briefs”
  - People who wrote these often knew each other, traded documents (we knew competitors would manage to get them anyway), asked for updated numbers, sometimes even traded code
  - Small, well-known benchmarks like Whetstone, Dhrystone, etc ... because we had to
  - More realistic larger codes, sometimes proprietary, sometimes open-sourced, but with different versions or inputs

# Example – MIPS Performance Brief 3.5, October 1988

---

- Most realistic benchmarks highlighted

|  |    |
|--|----|
| 1 Introduction .....                                   | 1  |
| 2 Benchmark Summary .....                              | 2  |
| 2.1 Choice of Benchmarks .....                         | 2  |
| 2.2 Benchmark Summary Data .....                       | 2  |
| 3 Methodology .....                                    | 4  |
| 4 Integer Benchmarks .....                             | 7  |
| 4.1 MIPS UNIX Benchmarks (MIPS UNIX) .....             | 7  |
| 4.2 Dhrystone (DHRV 1.1) .....                         | 10 |
| 4.3 Stanford Small Integer Benchmarks (STAN INT) ..... | 13 |
| 5 Floating Point Benchmarks .....                      | 15 |
| 5.1 Livermore Fortran Kernels (LLNL DP) .....          | 15 |
| 5.2 LINPACK (LNPK DP and LNPK SP) .....                | 19 |
| 5.3 Spice Benchmarks (SPCE 2G6) .....                  | 22 |
| 5.4 Digital Review .....                               | 24 |
| 5.5 Doduc Benchmark (DDUC) .....                       | 25 |
| 5.6 Whetstone .....                                    | 27 |
| 5.7 Stanford Floating Point Benchmarks .....           | 28 |
| 6 Acknowledgements .....                               | 29 |
| 7 References .....                                     | 29 |

# MIPS Performance Brief Summary, Geometric Mean

- 4 real Unix programs were used to publish aggregate integer performance
- Many quoted Dhrystone (VAX)-mips, ugh, but usually gave higher numbers than more realistic benchmarks
- Small benchmarks from Stanford
- Vendor-published mips ~consistent in own product line, but incommensurate
- Easier to find good floating point benchmarks than integer

## Summary of Benchmark Results Ugh

(VAX 11/780 = 1.0, Larger is Faster)

| Integer (C)  |             |             | Floating Point (FORTRAN) |            |            |             |      | Publ<br>mips | System                  |
|--------------|-------------|-------------|--------------------------|------------|------------|-------------|------|--------------|-------------------------|
| MIPS<br>UNIX | DHRY<br>1.1 | STAN<br>INT | LLNL<br>DP               | LNPk<br>DP | LNPk<br>SP | SPCE<br>2G6 | DDUC |              |                         |
| 1            | 1           | 1           | 1                        | 1          | 1          | 1           | 1    | 1            | VAX 11/780 <sup>#</sup> |
| 2.1          | 1.9         | 3.0         | 1.9                      | 2.9        | 2.5        | 1.6         | *1.3 | 2            | Sun-3/160 FPA           |
| *4           | 4.1         | 4.4         | 2.8                      | 3.3        | 3.4        | 2.4         | 1.7  | 4            | Sun-3/260 FPA           |
| 6.4          | 7.4         | 6.2         | 2.5                      | 4.3        | 3.7        | 3.4         | 3.8  | 5            | MIPS M/500              |
| *6           | 5.9         | 6.2         | 5.9                      | 7.1        | 5.6        | *5.3        | 5.2  | 6            | VAX 8700                |
| 9.9          | 10.8        | 10.7        | 4.5                      | 7.9        | 6.4        | 4.1         | 3.5  | 10           | Sun-4/200               |
| 10.9         | 11.3        | 10.0        | 8.1                      | 8.6        | 11.2       | 6.6         | 7.3  | 8            | MIPS M/800              |
| 13.1         | 13.5        | 12.3        | 10.8                     | 10.7       | 14.0       | 8.0         | 8.7  | 10           | MIPS M/1000             |
| 14.9         | 15.6        | 13.3        | 12.1                     | 15.0       | 16.0       | 9.7         | 11.1 | 12           | MIPS M/120-5            |
| 21.9         | 24.1        | 21.3        | 18.2                     | 25.7       | 23.6       | 16.0        | 17.0 | 20           | MIPS M/2000-8           |

Note: the Geometric Mean of N numbers is the Nth root of the product of those numbers. It is necessarily used in place of the arithmetic mean when computing the mean of performance ratios, or of benchmarks whose runtimes are quite different. See [Fleming 86] for a detailed discussion.

## MIPS UNIX Benchmarks Results

(4.3BSD VAX 11/780 = 1.0, Larger is Faster)

| grep |      | diff  |      | yacc  |      | nroff |      | Geom | System        | old-grep |      | old-nroff |      |
|------|------|-------|------|-------|------|-------|------|------|---------------|----------|------|-----------|------|
| Secs | Rel. | Sec   | Rel. | Secs  | Rel. | Secs  | Rel. | Mean |               | Secs     | Rel. | Secs      | Rel. |
| 58.5 | 1.0  | 246.4 | 1.0  | 101.1 | 1.0  | 108.1 | 1.0  | 1.0  | 11/780 4.3BSD | 11.2     | 1.0  | 18.8      | 1.0  |
| 29.2 | 2.0  | 105.3 | 2.3  | 48.1  | 2.1  | 51.5  | 2.1  | 2.1  | Sun-3/160M    | 5.6      | 2.0  | 9.0       | 2.1  |
| 7.8  | 7.5  | 35.8  | 6.9  | 19.5  | 5.2  | 17.5  | 6.2  | 6.4  | MIPS M/500    | 2.4      | 4.7  | 3.3       | 5.7  |
| 5.1  | 11.5 | 25.1  | 9.7  | 11.8  | 8.6  | 10.6  | 10.2 | 9.9  | Sun-4/200 -O3 | 1.6      | 7.0  | 2.2       | 8.6  |
| 5.1  | 11.5 | 21.7  | 11.4 | 11.2  | 9.0  | 9.2   | 11.8 | 10.9 | MIPS M/800    | 1.6      | 7.0  | 1.9       | 9.9  |
| 4.2  | 13.9 | 18.0  | 13.7 | 9.3   | 10.9 | 7.6   | 14.2 | 13.1 | MIPS M/1000   | 1.3      | 8.6  | 1.5       | 12.5 |
| 3.7  | 15.8 | 15.7  | 15.7 | 8.1   | 12.5 | 6.8   | 15.9 | 14.9 | MIPS M/120-5  | 1.1      | 10.2 | 1.3       | 14.5 |
| 2.5  | 23.4 | 11.3  | 21.8 | 5.4   | 18.7 | 4.5   | 24.0 | 21.9 | MIPS M/2000-8 | 0.7      | 16.0 | 0.9       | 20.9 |

# Origin of SPEC ~ October 1988 in Campbell, CA Bar

---

- Stan Baker prolific writer for widely-read *EE Times* magazine, wrote article with common “Dhrystone-MIPS”, i.e., performance relative to VAX-11/780
- I emailed him to complain about how awful that was (and I think another did)
- He responded: “If it’s so bad, why don’t you industry folks create something better?”  
**He offered his bar in Campbell, CA as a neutral meeting place, plus beer**
- I contacted folks from Hewlett-Packard, Sun Microsystems and Apollo Computer. All:
  - wanted to compete on realistic benchmarks than on poor ones like Dhrystone.
  - hated being asked by marketing to add features to speed Dhrystone, etc
  - disliked the amount of redundant effort, hard-to-compare results
  - encountered (legitimate) skepticism of customers confused by benchmark mess
- We wanted to codify best practices, often found in good performance briefs
  - Compile and run reasonably-portable source code for realistic C and Fortran programs. Real customer program were just not like popular, small, widely-quoted benchmarks
  - Clearly document system configurations, compiler flags
  - Every program needed verification test to assure correct execution
  - Clear reporting to the public
  - Customers liked consistent relative-performance ratios
  - Many of our documents included performance relative to VAX-11/780
- We knew we needed to get more vendors involved, sometimes over skepticism.

# A Year of Work to First Release

---

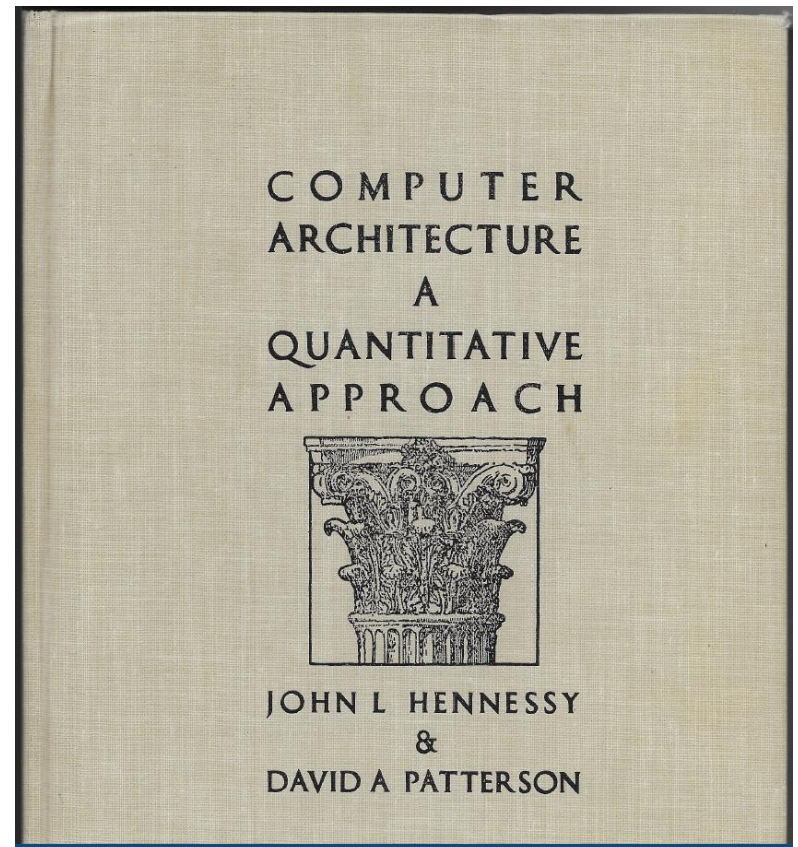
- Incorporated 11/14/88, with *EE Times*' Stan Baker as (neutral) President
- Year of hard work
  - Evaluate suggestions for good benchmarks
  - “Fuzzy compares” needed for some floating point codes, given differences
  - “Bench-a-thon” workshops, found surprising nonportabilities, difficulties
  - Many discussions over rules for running codes, validating results and reporting
  - Need to distribute source code ruled out proprietary programs  
University software eventually had turned into products, like spice→Hspice
- SPEC Release 1 announced Fall 1989 press conference, help from *EE Times*
  - 4 integer benchmarks, surprisingly hard to find good ones
  - 6 floating point benchmarks, many choices
  - 10 SPECratios, given as performance / VAX-11/780
  - SPECmark = SPEC89 = Geometric Mean (GM) of all 10
  - Wanted 2 separate summary numbers, but **got strong pushback from press**, reluctantly used one ... but vendors quickly published both
  - SPEC fixed 1992, deleted some, added more: (6) SPECint92, (10) SPECfp92
- A member of press asked fair question:
  - **“These are done by computer companies. Isn't that letting foxes guard the henhouse?”**
  - Our answer: **“Nobody is better than a fox at keeping other foxes from eating the hens.”**
  - **Coopetition** of fierce competitors can actually work well, once culture of trust and honesty built, and people find benefits from avoiding duplicative efforts.



# Incorporation In Famed Book Series

---

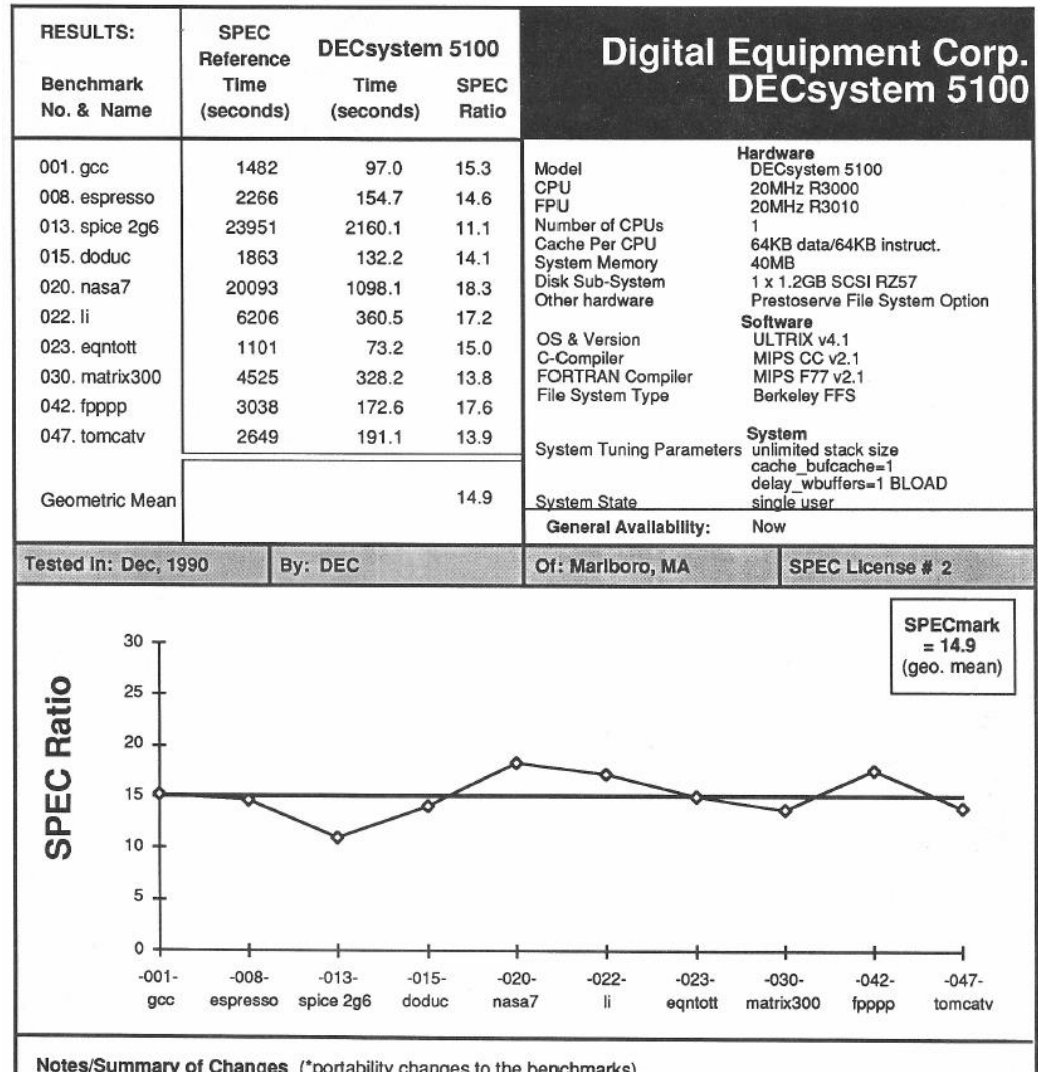
- John L. Hennessy and David A. Patterson  
Computer Architecture – A Quantitative Approach, Morgan Kaufmann, 1990.  
Rapidly became the standard high-level textbook.
- Later editions with serious revisions:  
1996, 2003, 2007, 2011, 2017.
- Essentially guaranteed SPEC visibility,  
analysis, discussion in universities →  
associate memberships



# Sample Report

- Standard reporting format, example from SPEC Newsletter Volume 3, Issue 1, Winter 1991.
- In Campbell bar, all liked Spice circuit simulator as big floating point benchmark, but we accidentally picked a poor choice of input.  
**Warning: no assumptions!**
- All liked Gnu c (gcc) benchmark only one still in SPEC CPU2017 (C, C++, Fortran)  
10 integer  
13 floating point

<https://www.spec.org/cpu2017/Docs/overview.html#benchmarks>





# March 1991 HP Performance Brief

- During 1990-1991, vendors were including SPEC, for example:  
HP Apollo 9000 Series 700  
Performance Brief March 1991
- matrix300 leap?**  
In 1991, several vendors started to enhance their Fortran compilers to use method called “cache-blocking” to improve vector/matrix codes. It was dropped in 1992
- Valuable, legitimate optimization, but too small and simple, was sped up far more than seen on more realistic codes.

Table 1.1.1 SPEC Benchmarks

The integer (non-floating point) benchmarks are:

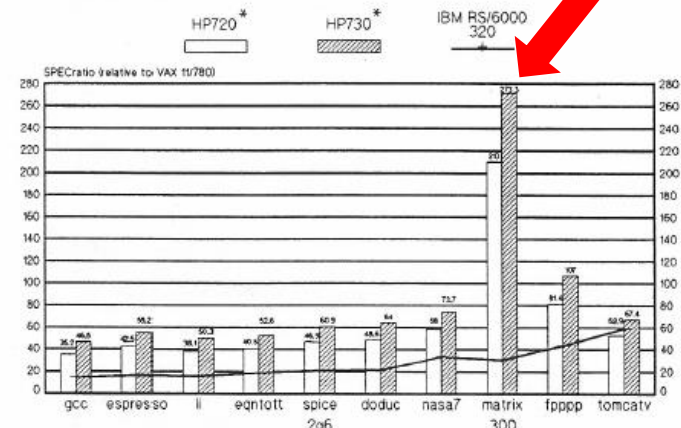
- **gcc** - based on the GNU C compiler version 1.35 distributed by the Free Software Foundation. This benchmark measures the time it takes for the GNU C compiler to convert 19 pre-processed source files into optimized Sun-3 assembly language output.
- **espresso** - one of a collection of tools for the generation and optimization of Programmable Logic Arrays (PLAs). Written in C this benchmark is representative of many EDA applications.
- **li** - a LISP interpreter written in C. This benchmark measures the time required for li to solve the 9-queens problem
- **eqntott** - written in C, translates a logical representation of a boolean equation to a truth table.

The floating point benchmarks are:

- **spice2g6** - an analog circuit simulation and analysis application that is heavily used in EE CAD. Spice is written in FORTRAN.
- **doduc** - a non-vectorizable, scalar FORTRAN benchmark. It is a Monte Carlo simulation.
- **nasa7** - a collection of seven FORTRAN kernels.
- **matrix300** - a vectorizable FORTRAN scientific benchmark. This code performs various matrix multiplications.
- **fpppp** - a quantum chemistry benchmark written in FORTRAN.
- **tomcatv** - a vectorized mesh generation FORTRAN program, tomcatv is a part of Prof. W. Gentzsch's benchmark suite.

Figure 1.1.3

SPEC Individual Benchmark Results  
HP 700 & IBM RS/6000 320



\* with HP UX and compilers available June '91

# SPEC Membership, Winter 1991

- SPEC Newsletter, Volume 3, Issue 1, Winter 1991  
Most major vendors had signed up...
- Individual vendors had often been reporting summaries separately for integer and floating point subsets.
- By SPEC 1992, they were officially split.
- All this was pre-Internet.  
Results were printed in Newsletters.  
Benchmark codes were shipped on tapes.

Arix Corp.  
AT&T  
Bull S.A.  
Compaq Computer Corp.  
Control Data Corporation  
Data General Corporation  
Digital Equipment Corporation  
Fujitsu Ltd.  
Hewlett-Packard  
IBM  
Intel Corporation  
Intergraph Corp.  
MIPS Computer Systems  
Motorola  
NCR Corporation  
Prime Computer  
Siemens Nixdorf  
Silicon Graphics  
Solbourne Computer  
Stardent Computer  
Sun Microsystems  
Unisys Corporation

# Explanations – VAX as (first) Reference Machine

---

- SPEC Newsletter, Volume 2, Issue 1, Winter 1990, by Kaivalya Dixit (Sun) and Robert Novak (MIPS)

## “VAX As Reference Machine

Since SPEC first began publishing results, questions about the use of the VAX-11/780 as the reference machine have arisen. The VAX was chosen because it is generally recognized as a 1 mip (millions of instructions per second) machine. With its large installed base, the VAX provides a better understanding of the relative performance of machines in regards to the Benchmark Suite. The reference machine issue can be likened to a barometer that reports numbers that are worthless unless the mean barometric pressure is known. However, there is no guarantee that SPEC will use the VAX as the reference platform for future releases. As the benchmarks become more complex, SPEC may switch to a more appropriate reference machine.”

*Indeed, newer systems got so much faster it took far too long to run the programs on old VAX-11/780s, and by 1992 the industry was accustomed to the methodology.*

*It computed consistent relevant performance summaries regardless of choice of reference machine by:*

*Converting run-times to relative performance= SPECratios*

*Computing Geometric Means from SPECratios*

*That was important: there could be no advantage/disadvantage from being chosen as reference.*

*However, in the 1980s and even later, the choice of summary computation was very contentious among Arithmetic Mean, Harmonic Mean and Geometric Mean. (Livermore Loops reported all 3.)*

*Many papers were written.*

# Geometric Mean and Statistics

- If well-characterized workload, one can just run the exact programs and measure results, sometimes true for embedded systems and occasionally in high-performance computing
- Otherwise, a benchmark set (like SPEC) should be treated as a sample of population: If chosen by experts, it may be representative sample
- Statistical methods should be applied. For many data sets:
  - Compute (arithmetic) mean (AM), standard deviation, skew, kurtosis (1<sup>st</sup> 4 moments)
  - Hope for normality (Gaussian) for its properties, but test, as never guaranteed
- But for sets of SPECratios, AM gives contradictory results.  
Extreme example: system A is no better than machine B on combination of benchmarks 1 and 2, but A can seem 1.25X faster if B is chosen as reference, but arbitrary choice. 1980s “ratio games”  
GM gives consistent results, although our charts should have been logarithmic

| Sys | Benchmarks |           | Relative Perf |          | AM   | GM  |
|-----|------------|-----------|---------------|----------|------|-----|
|     | 1<br>Perf  | 2<br>Perf | 1<br>A/B      | 2<br>A/B |      |     |
| A   | 2          | 1         | 2.0           | 0.5      | 1.25 | 1.0 |
| B   | 1          | 2         | 1.0           | 1.0      | 1.0  | 1.0 |

| Sys | Benchmarks |           | Relative Perf |          | AM   | GM  |
|-----|------------|-----------|---------------|----------|------|-----|
|     | 1<br>Perf  | 2<br>Perf | 1<br>B/A      | 2<br>B/A |      |     |
| A   | 2          | 1         | 1.0           | 1.0      | 1.0  | 1.0 |
| B   | 1          | 2         | 0.5           | 2.0      | 1.25 | 1.0 |

# Geometric Mean, Statistics, Lognormal Distribution

- Many areas of science: people use logarithmic transforms, often hoping to find a **lognormal distribution** (generated by combinations of multiplicative effects)
- In 1989 we missed a basic mathematical identity.  
Usual way to compute GM was just shorthand for exponentiating average of logarithms of data
$$GM = \overline{x_G} = \left( \prod_{i=1}^n x_i \right)^{\left(\frac{1}{n}\right)} = \exp\left(\frac{1}{n} \sum_{i=1}^n \ln(x_i)\right)$$
- Then, one can do standard statistics:  
Compute mean, standard deviation, skew, kurtosis of logs  
Perform normality tests, if logs are normal or close, distribution is ~lognormal.
  - $GM(y:x) = \exp(\text{mean of logs})$
  - Standard deviation =  $\exp(\log \text{ standard deviation})$  = multiplicative standard deviation sigma, i.e., if lognormal and sigma = 1.2, ~2/3 of data should lie within  $[GM/1.2 \dots GM*1.2]$
  - $GM(y:x) = 1/GM(x:y)$  better be
  - $\text{Sigma}(y:x) = \text{sigma}(x:y)$  invariant to choice of base.
  - $\text{Skew}(y:x) = -\text{skew}(x:y)$  reverses left-to-right
  - $\text{Kurtosis}(y:x) = \text{Kurtosis}(x:y)$  has same shape
- J. R. Mashey, "War of the benchmark means: time for a truce," ACM SIGARCH Computer Architecture News vol 32, issue 4, pp.1-14, September 2004. Doi: 10.1145/1040136.1040137  
J. R. Mashey, "Summarizing performance is no mean feat," October 30, 2008, talk given 2005-2008 at various places. <https://techviser.com/docs/Mashey.nomeanfeat.2008.pdf> Many examples, 77p.  
[https://en.wikipedia.org/wiki/Log-normal\\_distribution](https://en.wikipedia.org/wiki/Log-normal_distribution)
- **Standard statistical methods can be used to analyze distributions, not just averages.**

# Lessons - NLP(?)

---

- Knowing ~nothing of NLP benchmarking, I looked around
- Metrics? Accuracy, Precision, Recall or F1?  
<https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9>
- Dynabench: Rethinking Benchmarking in NLP  
<https://arxiv.org/abs/2104.14337>
- <https://paperswithcode.com/>
- Natural Language Processing Performance Metrics (Benchmarks)  
<https://dev.to/amananandrai/natural-language-processing-performance-metrics-benchmarks-4jel>  
GLUE, SuperGLUE, SQuAD, BLEU, MS MACROS, XTREME
- This is obviously a huge space, some applications seem to overlap with MLCommons, but there may be generic lessons from experience with SPEC and others.
- GLUE/SuperGLUE looks like promising start. There may be more.



# GLUE, SuperGLUE?

---

- GLUE: A MULTI-TASK BENCHMARK AND ANALYSIS PLATFORM FOR NATURAL LANGUAGE UNDERSTANDING <https://arxiv.org/pdf/1804.07461.pdf>
- SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems <https://super.gluebenchmark.com/>  
<https://w4ngatang.github.io/static/papers/superglue.pdf>

“GLUE is a collection of nine language understanding tasks built on existing public datasets, together with private test data, an evaluation server, a single-number target metric, and an accompanying expert constructed diagnostic set. ...The **progress of the last twelve months has eroded headroom** on the GLUE benchmark dramatically. While some tasks (Figure 1) and some linguistic phenomena (Figure 2 in Appendix B) measured in GLUE remain difficult, the current state of the art GLUE Score as of early July 2019 (88.4 from Yang et al., 2019) surpasses human performance (87.1 from Nangia and Bowman, 2019) ... Consequently, while there remains substantial scope for improvement towards GLUE’s high-level goals, the original version of the benchmark is no longer a suitable metric for quantifying such progress. ... In response, we introduce SuperGLUE, a new benchmark designed to pose a more rigorous test of language understanding. SuperGLUE has the same high-level motivation as GLUE: **to provide a simple, hard-to-game measure of progress** toward general-purpose language understanding technologies for English.”

  - **8 benchmark test sets, average for single figure of merit**
  - **Reminiscent of SPEC in early days!**

# GLUE, SuperGLUE

- SuperGLUE paper uses numbers rescaled to human performance = 1 (VAX-11/780 in SPECmark89!)
- Blue = GLUE overall score ... but is that (correct) GM of rescaled numbers or (wrong) AM?
- “average of multiple metrics” seems OK if commensurate, otherwise may skew distribution or have a stronger effect than expected on overall result. If numbers close, may not matter.

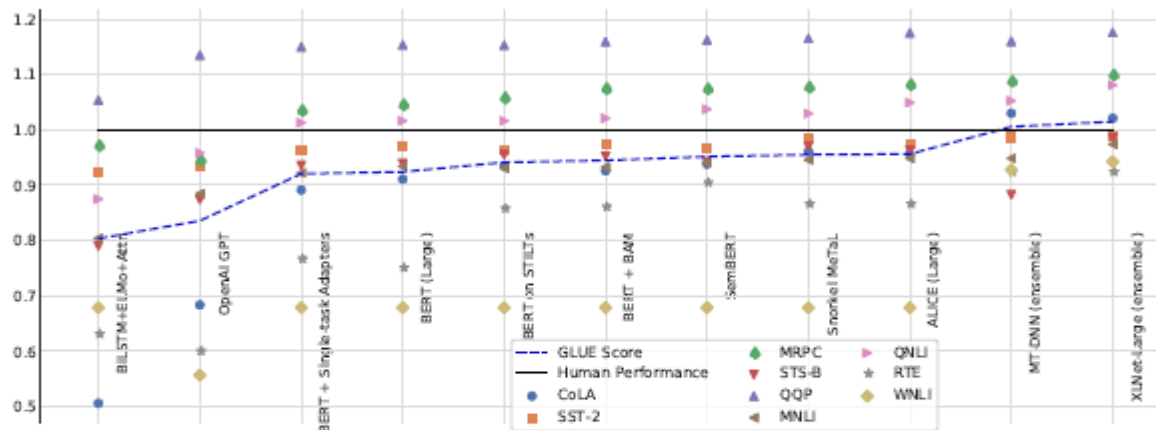


Figure 1: GLUE benchmark performance for submitted systems, rescaled to set human performance to 1.0, shown as a single number score, and broken down into the nine constituent task performances. For tasks with multiple metrics, we use an average of the metrics. More information on the tasks included in GLUE can be found in Wang et al. (2019a) and in Warstadt et al. (2018, CoLA), Socher et al. (2013, SST-2), Dolan and Brockett (2005, MRPC), Cer et al. (2017, STS-B), and Williams et al. (2018, MNLI), and Rajpurkar et al. (2016, the original data source for QNLI).



# SuperGLUE, Leaderboard

- <https://super.gluebenchmark.com/leaderboard>
- ERNIE 3.0: Large-scale Knowledge Enhanced Pre-training for Language Understanding and Generation <https://arxiv.org/abs/2107.02137>  
Many benchmarks, reminiscent of 1980s Performance Briefs
- P.14 top 4 from leaderboard, plus a few extra statistics  
Note NLU software advantage over Human is primarily due to poor results on MultiRC.  
Human is actually best on 4 of 8 benchmarks. Care needed to avoid over-interpretation.  
If MRC dropped, Human is ahead.

| Model          | BoolQ | CB        | COPA | MultiRC   | ReCoRD    | RTE  | WiC  | WSC  | Score | Stddev | Skew | Kurtosis | Score | Score-MRC |
|----------------|-------|-----------|------|-----------|-----------|------|------|------|-------|--------|------|----------|-------|-----------|
| Human Baseline | 89.0  | 95.8/98.9 | 100  | 81.8/51.9 | 91.7/91.3 | 93.6 | 80.0 | 100  | 89.8  | 11.4   | -1.3 | 1.5      | 89.8  | 93.1      |
| T5+Menna       | 91.4  | 95.8/97.6 | 98.0 | 88.3/63.0 | 94.2/93.5 | 93.0 | 77.9 | 96.6 | 90.4  | 8.8    | -1.2 | -0.3     | 90.5  | 92.6      |
| DeBERTa        | 90.4  | 95.7/97.6 | 98.4 | 88.2/63.7 | 94.5/94.1 | 93.2 | 77.5 | 95.9 | 90.3  | 8.7    | -1.1 | -0.4     | 90.3  | 92.3      |
| ERNIE 3.0      | 91.0  | 98.6/99.2 | 97.4 | 88.6/63.2 | 94.7/94.2 | 92.6 | 77.4 | 97.3 | 90.6  | 9.0    | -1.1 | -0.4     | 90.6  | 92.7      |

# Lessons: 1989 (CPU) : 2021 (NLP)

---

- In fast-moving field, plethora of benchmarks very likely ... and likely confusing
- Successful benchmark efforts possible, but take work & persistence
  - Coopetition works, if people are reasonable and trust develops
  - Start small and focused, develop extendable processes
  - Later, expand to adjacent areas
  - Sometimes act as umbrellas, other groups may join, as GPC and PERFECT → SPEC
  - But no one group can do it all, depends on structure of the domain
- Industry / academe (and often government) cooperation important
- Individual benchmarks vs summaries
  - People must relate to the individual benchmarks or sets of benchmarks, maybe ignore rest
  - But many also want summary figures
  - **Hope: more analysis of distributions, not just averages, avoid over-interpretation**
- Good statistical methods should be applied widely to calibrate depth of knowledge
  - Sometimes multi-modal distributions hint that several populations are mixed
  - Outliers can be instructive, may strongly affect summaries
- From outside, SuperGLUE seems a credible effort to collect and organize results, obviously more benchmarks desirable for better statistics, coverage