



# **High Performance Computer Architecture (CS60003)**

**Dept. of Computer Science & Engineering  
Indian Institute of Technology Kharagpur**

**Spring 2020**



# Benchmarks



# Performance Evaluation

- Rely on Queuing theoretical or analytical models: When the underlying system can be described as an abstract model.
  - Example: Assess the throughput of a disk subsystem when the input rate and service time can be expressed as statistical distributions.
  - However, this is not in our scope.
- Measure the execution time of a suite of programs, called benchmarks.
- Simulate the whole system or components.

# Benchmarks

- We intend to study the general purpose micro-processor machines.
- The choice of the best machine depends on the workload they perform
  - Single user: That which executes the common programs fastest.
  - Manager of a Webserver: That with least mean response time for queries on the web.
  - Designer of a database: Throughput of transactions.

Benchmarks, are a suite of test programs, available for each category of users.

When we are interested to evaluate the processor and the memory hierarchy, we look forward to those which evaluate in a compute-intensive manner. - neglecting, OS, Networking, IO effects.

# Benchmark Classifications

## ■ Synthetic Benchmarks:

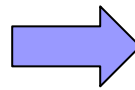
- Artificial programs specially written to measure performance of specific architectural factors.
- Example1, Whetstone: contains a very large proportion of floating point instructions.
- Example1, Dhrystone: contains a very large proportion of string operations.
- Not used for modern processors, but used for embedded processors.

# Benchmark Classifications

## ■ Kernels:

- Small portions of real programs that exercise a certain feature of the system.
- Example1: Livermore. It contains 24 FORTRAN DO loops extracted from real programs.
  - Used for testing potential vectorization of the loops themselves, whether by hardware or by compilers.
    - ***Vectorization is the process of converting an algorithm from operating on a single value at a time to operating on a set of values at one time.***  
Modern CPUs provide direct support for vector operations where a single instruction is applied to multiple data (SIMD).

```
for (int i=0; i<16; ++i)  
C[i] = A[i] + B[i];
```



```
for (int i=0; i<16; i+=4)  
C[i:i+3] = A[i:i+3]+B[i:i+3];
```

- Microbenchmarks: Isolate & measure special characteristics of architecture.

# Benchmark Classifications

- Complete Programs:
  - Suite of selected programs depends on the intended use of the benchmarks.
  - Nonprofit organization of SPEC (System Performance Evaluation Corporation) tries to “establish, maintain and endorse a standardized set of relevant benchmarks that can be applied to the newest generation of high-performance computers”
  - Publishes benchmarks for variety of applications
    - SPEC CP XXXX: Most relevant to microprocessor vendors and designers (XXXX is the year, like 2000)
    - Olden Suite: 10 programs with extensive linked list processing, thus offering challenges to exploitation of instruction level parallelism and to hide memory latencies.
    - TPC (Transaction Processing Council): Collection of programs for commercial workloads to evaluate servers.

# SPEC Benchmarks

- Announced in 1989
- Expressed the performance of a system relative to that of a VAX 11/780, which had a SPEC rating of 1.
- It was modified in 92, 95.
- It was measured relative to a new machine, a 40MHz Sun SPARCstation without a 2nd level cache, and 64MB of memory.
- Revised again in 2000, SPEC2000.
- Normalizing machine was a Sun Ultra5\_10 SparcStation, that run at 300 MHz , 2 level cache hierarchy and 256MB of memory. SPEC Rating of 100.
- Revised again in 2006, similar processor but larger cache and main memory. Recent in 2017 (<https://www.spec.org/cpu2017>)



# Ingredients

- CPU2000 consists of 12 integer programs (11 written in C, 1 in C++), 14-floating point programs (11 written in FORTRAN77 or 90, and 3 in C)
- Some examples, Compression, FPGA circuit placement and routing, Game playing CHESS, Group Theory interpreter, quantum, water modeling, 3D-graphics, seismic wave, primality testing, finite element car crash, image processing, meteorology, word processing, etc.
- Each program has 3 datasets:
  - test: quick functional test
  - train: medium length runs
  - reference: one used for reporting results
    - Runs for tens or hundreds of billions of instructions.
    - 3 GHz machine with an IPC of 1 it would take 10 seconds to run 30 billion instruction benchmark
    - Compiler optimizations which are specific to any program are disallowed.

# Is SPEC representative?

- There seems to be only one program mildly representative of desktop applications (parser)
- Not present in 2006
- There is only one interpreter program-perl
- Some C++ programs in 2006, but no Java programs.
- However, studies of desktop applications and interpreters have shown that:
  - their instruction mixes and,
  - the way they exercise the lower levels of the memory hierarchyare pretty similar to what is happening with the integer programs in SPEC benchmarks.

# Reporting Benchmark Results

- Several ways have been advocated.
- Most obvious way is to report the average as arithmetic mean, ie.  $T = \frac{1}{n} \sum_i T_i$
- If some programs are more important than others we computed weighted arithmetic means.
- If instead of measuring the rates, like IPC, we use harmonic mean.

□ Thus,  $IPC = \frac{n}{\sum_i \frac{1}{IPC_i}}$

# Reporting Benchmark Results

- SPEC does not use arithmetic means of execution times
- It normalizes the results of the benchmarks to some reference machine.
- Then the geometric means of the ratios of the execution time of the runs are reported.
  - $G = \sqrt[n]{\prod_i \frac{T_i}{S_i}}$ ,  $T_i$  is the execution time for the  $i^{th}$  program, and  $S_i$  is the execution time on the reference machine.
  - One advantage, if comparisons between 2 machines be made wrt. an old reference machine, no reevaluation needs to be done wrt. a new reference machine.

# Metrics can fool!

	Normalization time on reference machine	Normalized time on M1	Normalized time on M2
Program P1	1	5	10
Program P2	1	5	2
Arithmetic Mean	1	5	6
Geometric Mean	1	5	4.5

Total time for M1=10 time units, for M2=12.

However, Geometric Mean shows M2 is faster!


See how the benchmark can be manipulated.

If the time for M2 for P2 was 3, Geometric Mean for M2 would have been higher.

By “optimizing” the short program by 1 time unit, shows anomaly.

However, opportunities for such optimizations does not exist in SPEC benchmarks, because number of instructions executed in each program is of the same order.

# A Sample Report

 <b>SPEC® CPU2017 Floating Point Rate Result</b> <small>Copyright 2017-2018 Standard Performance Evaluation Corporation</small>	
ASUSTeK Computer Inc. ASUS RS700-E9(Z11PP-D24) Server System (2.70 GHz, Intel Xeon Gold 6150)	SPECrate2017_fp_base = 199 SPECrate2017_fp_peak = 201
CPU2017 License: 9016 Test Sponsor: ASUSTeK Computer Inc. Tested by: ASUSTeK Computer Inc.	Test Date: Dec-2017 Hardware Availability: Jul-2017 Software Availability: Sep-2017

Benchmark result graphs are available in the PDF report.

Hardware		Software	
<b>CPU Name:</b>	Intel Xeon Gold 6150	<b>OS:</b>	SUSE Linux Enterprise Server 12 (x86_64) SP2
<b>Max MHz.:</b>	3700		Kernel 4.4.21-69-default
<b>Nominal:</b>	2700	<b>Compiler:</b>	C/C++: Version 18.0.0.128 of Intel C/C++ Compiler for Linux; Fortran: Version 18.0.0.128 of Intel Fortran Compiler for Linux
<b>Enabled:</b>	36 cores, 2 chips, 2 threads/core	<b>Parallel:</b>	No
<b>Orderable:</b>	1, 2 chip(s)	<b>Firmware:</b>	Version 0601 released Oct-2017
<b>Cache L1:</b>	32 KB I + 32 KB D on chip per core	<b>File System:</b>	btfs
<b>L2:</b>	1 MB I+D on chip per core	<b>System State:</b>	Run level 3 (multi-user)
<b>L3:</b>	24.75 MB I+D on chip per chip	<b>Base Pointers:</b>	64-bit
<b>Other:</b>	None	<b>Peak Pointers:</b>	64-bit
<b>Memory:</b>	768 GB (24 x 32 GB 2Rx4 PC4-2666V-R)	<b>Other:</b>	None
<b>Storage:</b>	1 x 960 GB SATA SSD		
<b>Other:</b>	None		

## Results Table

Benchmark	Base							Peak						
	Copies	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio	Copies	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio
503.bwaves_r	72	<b>1529</b>	<b>472</b>	1529	472	1530	472	72	1526	473	1526	473	<b>1526</b>	<b>473</b>
507.cactuBSSN_r	72	504	181	<b>505</b>	<b>181</b>	505	180	72	<b>505</b>	<b>180</b>	505	180	505	180
508.namd_r	72	383	179	381	180	<b>383</b>	<b>179</b>	72	382	179	<b>382</b>	<b>179</b>	383	179
510.parest_r	72	1669	113	<b>1684</b>	<b>112</b>	1700	111	72	1687	112	1699	111	<b>1695</b>	<b>111</b>
511.povray_r	72	<b>603</b>	<b>279</b>	618	272	596	282	72	521	323	<b>524</b>	<b>321</b>	525	320
519.lbm_r	72	678	112	677	112	<b>677</b>	<b>112</b>	72	<b>677</b>	<b>112</b>	678	112	677	112
521.wrf_r	72	772	209	<b>784</b>	<b>206</b>	787	205	72	788	205	785	205	<b>785</b>	<b>205</b>
526.blender_r	72	<b>495</b>	<b>222</b>	495	222	495	222	72	487	225	<b>487</b>	<b>225</b>	488	225
527.cam4_r	72	547	230	<b>550</b>	<b>229</b>	550	229	72	543	232	<b>542</b>	<b>232</b>	542	232
538.imagick_r	72	509	352	<b>509</b>	<b>352</b>	510	351	72	511	351	<b>511</b>	<b>351</b>	511	351
544.nab_r	72	<b>399</b>	<b>303</b>	398	304	400	303	72	396	306	<b>396</b>	<b>306</b>	395	307
549.fotonik3d_r	72	1985	141	<b>1986</b>	<b>141</b>	1986	141	72	<b>1985</b>	<b>141</b>	1981	142	1985	141
554.roms_r	72	1263	90.6	<b>1268</b>	<b>90.2</b>	1269	90.2	72	1271	90.0	1270	90.1	<b>1271</b>	<b>90.0</b>
SPECrate2017_fp_base		199												
SPECrate2017_fp_peak		201												

Results appear in the order in which they were run. Bold underlined text indicates a median measurement.









**Thank You!**