性能评价

郭健美

2023年秋

SOLE 系统优化实验室
华东师范大学

A Systematic Approach to Performance Evaluation

1. State Goals and Define the System
2. List Services and Outcomes
3. Select Metrics
4. List Parameters
5. Select Factors to Study
6. Select Evaluation Technique
7. Select Workload
8. Design Experiments
9. Analyze and Interpret Data
10. Present Results

Repeat

A Systematic Approach to Performance Evaluation

1. Define goals (systems, services, and outcomes)
2. Select evaluation methods
3. Select metrics
4. Select workloads
5. Design experiments (parameters, factors)
6. Analyze and interpret data
7. Present results

Repeat

A Systematic Approach to Performance Evaluation

1. **Define goals (systems, services, and outcomes)**
2. Select evaluation methods
3. Select metrics
4. Select workloads
5. Design experiments (parameters, factors)
6. Analyze and interpret data
7. Present results

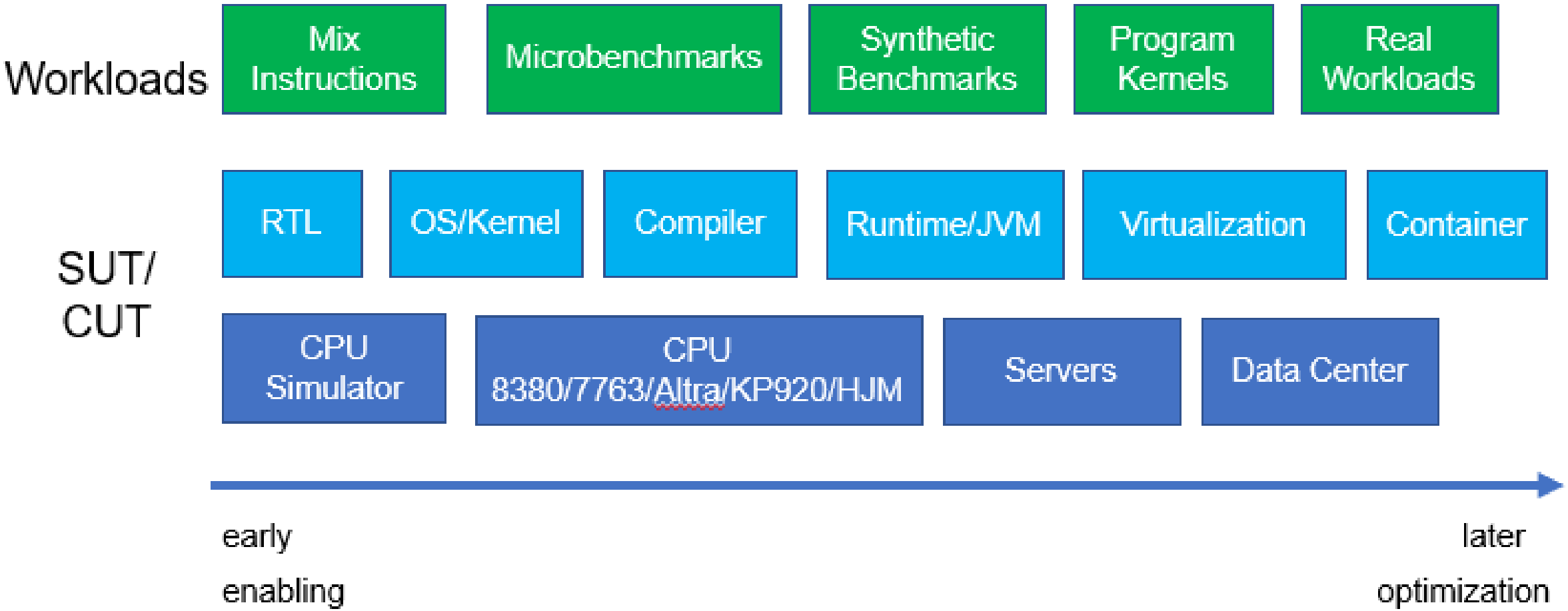
Repeat

State Goals

- Identify the goal of study
  - Not trivial, but
  - Will affect every decision or choice you make down the road
- Clearly define the system
  - Where you draw the boundary will
    - Dictate the choice of model
    - Affect choice of metrics and workload

List Services and Outcomes

- Identify the services offered by the system
- For each service, identify all possible outcomes
- What's the point
  - These will help in the selection of appropriate metrics


Workloads
Mix Instructions
Microbenchmarks
Synthetic Benchmarks
Program Kernels
Real Workloads
SUT/ CUT
RTL
OS/Kernel
Compiler
Runtime/JVM
Virtualization
Container
CPU Simulator
CPU 8380/7763/Altra/KP920/HJM
Servers
Data Center
early
later
enabling
optimization

A Systematic Approach to Performance Evaluation

1. Define goals (systems, services, and outcomes)
2. **Select evaluation methods**
3. Select metrics
4. Select workloads
5. Design experiments (parameters, factors)
6. Analyze and interpret data
7. Present results

Repeat

Criteria for Selecting an Evaluation Technique

Criterion	Analytical Modeling	Simulation	Measurement
1. Stage	Any	Any	Postprototype
2. Time required	Small	Medium	Varies
3. Tools	Analysts	Computer languages	Instrumentation
4. Accuracy[a]	Low	Moderate	Varies
5. Trade-off evaluation	Easy	Moderate	Difficult
6. Cost	Small	Medium	High
7. Saleability	Low	Medium	High

[a] In all cases, result may be misleading or wrong.

Measurement, Simulation, or Analysis?

- Can be a combination of two or all three
- Use the goal of study to guide your decision
- The resources and skills available may also be taken into account
- Remember, each of these techniques has its pros and cons
  - Let us look at some of them

Measurement

- (+) Provides realistic data
- (+) Can test the limits on load
- (-) System or a prototype should be working
- (-) The prototype may not represent the actual system
- (-) Not that easy to correlate cause and effect
- Challenges
  - Defining appropriate metrics
  - Using appropriate workload
  - Statistical tools to analyze the data

Simulation

- (+) Less expensive than building a prototype
- (+) Can test under more load scenarios
- (-) Synthetic since the model is not the actual system
- (-) Can not use simulation to make any guarantees on expected performance
- Challenges
  - Need to be careful when to use simulation
  - Need to get the model right
  - Need to represent results well (the graphical tools)
  - Need to learn simulation tools

Analytical Modeling

- (+) Can make strong guarantees on expected behavior
- (+) Can provide an insight in to cause and effect
- (+) Does not need to build a prototype
- (-) Performance prediction only as good as the model
- Challenges
  - Significant learning curve
  - Mathematically involved
  - Choosing the right model (the art work)

Bottom Line

- You can use measurement to demonstrate feasibility of an approach.
- You can use measurement or simulation to show an evidence that your algorithm or system performs better than competing approaches in certain situations.
- But, if you would like to claim any properties of your algorithm (or system), the only option is to use analysis and mathematically prove your claim.

When to Use What?

- ❑ It is good to be versed in all three

1. Can start with measurement or simulation to get a feel of the model or expected behavior
2. Start with a simple model
3. Perform an analysis to predict the performance and prove some behavioral properties
4. Observe the actual performance to determine the validity of your model and your analysis
5. Can use simulation for the previous step if a working system is not available/feasible
6. Go back to revise the model and analysis if significant inconsistency is observed and start with Step 4
7. Finally use simulation to verify your results for large scale data or for scenarios that can not be modeled with existing expertise and available time

Three Rules of Validation

- Do not trust the results of a **simulation model** until they have been validated by analytical modeling or measurements.
- Do not trust the results of an **analytical model** until they have been validated by a simulation model or measurements.
- Do not trust the results of a **measurement** until they have been validated by simulation or analytical modeling.

A Systematic Approach to Performance Evaluation

1. Define goals (systems, services, and outcomes)
2. Select evaluation methods
3. **Select metrics**
4. Select workloads
5. Design experiments (parameters, factors)
6. Analyze and interpret data
7. Present results

Repeat

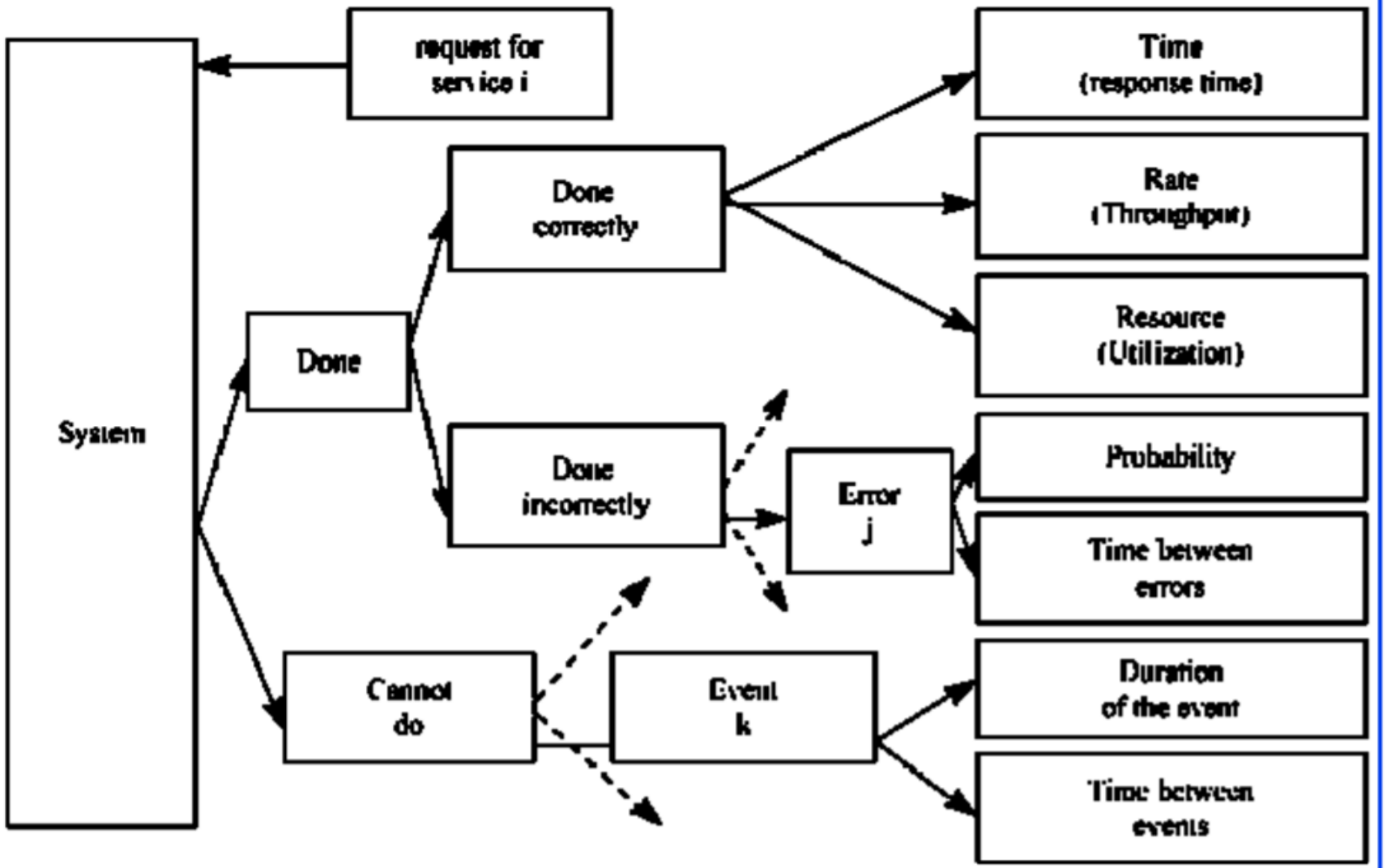
SPEC CPU 2017 Metrics

- There are many ways to measure computer performance. Among the most common are:
  - Time - For example, seconds to complete a workload.
  - Throughput - Work completed per unit of time, for example, jobs per hour.
- SPECspeed is a time-based metric; SPECrate is a throughput metric.

Calculating SPECspeed® Metrics	Calculating SPECrate® Metrics
1 copy of each benchmark in a suite is run.	The tester chooses how many concurrent copies to run
The tester may choose how many OpenMP threads to use.	OpenMP is disabled.
For each benchmark, a performance ratio is calculated as: `time on a reference machine / time on the SUT`	For each benchmark, a performance ratio is calculated as: `number of copies * (time on a reference machine / time on the SUT)`
Higher scores mean that less time is needed.	Higher scores mean that more work is done per unit of time.
Example: • The reference machine ran 600.perlbench_s in 1775 seconds. • A particular SUT took about 1/5 the time, scoring about 5. • More precisely: `1775/354.329738 = 5.009458`	Example: • The reference machine ran 1 copy of 500.perlbench_r in 1592 seconds. • A particular SUT ran 8 copies in about 1/3 the time, scoring about 24. • More precisely: `8*(1592/541.52471) = 23.518776`
For both SPECspeed and SPECrate, in order to provide some assurance that results are repeatable, the entire process is repeated. The tester may choose: a. To run the suite of benchmarks three times, in which case the tools select the medians. b. Or to run twice, in which case the tools select the lower ratios (i.e. slower).	
For both SPECspeed and SPECrate, the selected ratios are averaged using the Geometric Mean, which is reported as the overall metric.	

https://spec.org/cpu2017/Docs/overview.html

Selecting Performance Metrics



Washington University in St. Louis CSE567M ©2008 Raj Jain

Selecting Metrics

- Include:
  - Performance Time, Rate, Resource
  - Error rate, probability
  - Time to failure and duration
- Consider including:
  - Mean and variance
  - Individual and Global
- Selection Criteria:
  - Low-variability
  - Non-redundancy
  - Completeness

Criteria for Metric Set Selection

- Low-variability
  - Helps reduce the number of runs needed
  - Advice: Avoid ratios of two variables
- Non-redundancy
  - Helps make results less confusing and reduce the effort
  - Try to find a relationship between metrics
    - If a simple relationship exists, keep only one
- Completeness

Common Metrics

- Response Time
  - Turnaround time, reaction time
  - Stretch factor
    - Response time at a particular load divided by response time at minimum load
- Throughput
  - Nominal capacity: Under ideal workload
  - Usable capacity: With acceptable response time
- Efficiency: usable capacity/nominal capacity
- Utilization: busy time/elapsed time

其他注意事项

- 量纲分析 dimensional analysis：check if the metrics do have physical meanings
- 完整性检查 sanity check: check the boundary conditions (i.e. best system, ideal workload, etc.) to see if the metric is sensible

A Systematic Approach to Performance Evaluation

1. Define goals (systems, services, and outcomes)
2. Select evaluation methods
3. Select metrics
4. **Select workloads：L4 基准评测程序**
5. Design experiments (parameters, factors)
6. Analyze and interpret data
7. Present results

Repeat

A Systematic Approach to Performance Evaluation

1. Define goals (systems, services, and outcomes)
2. Select evaluation methods
3. Select metrics
4. Select workloads
5. **Design experiments (parameters, factors)：L3 实验设计**
6. Analyze and interpret data
7. Present results

Repeat

A Systematic Approach to Performance Evaluation

1. Define goals (systems, services, and outcomes)
2. Select evaluation methods
3. Select metrics
4. Select workloads
5. Design experiments (parameters, factors)
6. **Analyze and interpret data**
7. Present results

Repeat

Problem

Suppose that you measure the performance of a deterministic program 100 times on a computer with some interfering background noise. What statistic best represents the raw performance of the software?

- ❑ arithmetic mean
- ❑ geometric mean
- ❑ median
- ❑ maximum
- ❑ minimum

Problem

Suppose that you measure the performance of a deterministic program 100 times on a computer with some interfering background noise. What statistic best represents the raw performance of the software?

- ❑ arithmetic mean
- ❑ geometric mean
- ❑ median
- ❑ maximum
- ☑ minimum

Minimum does the best at noise rejection, because we expect that any measurements higher than the minimum are due to noise.

Selecting among Summary Statistics

Service as many requests as possible

- Arithmetic mean
- CPU utilization

All tasks are completed within 10 ms

- Arithmetic mean
- Wall-clock time

Most service requests are satisfied within 100 ms

- 90th percentile
- Wall clock time

Meet a customer service-level agreement (SLA)

- Some weighted combination
- multiple

Fit into a machine with 100 MB of memory

- Maximum
- Memory use

Least cost possible

- Arithmetic mean
- Energy use or CPU utilization

Fastest/biggest/best solution

- Arithmetic mean
- Speedup of wall clock time

Ratio vs Rate

Ratio vs Rate

- A **ratio** is a comparison of two numbers or measurements. The numbers or measurements being compared are sometimes called the **terms** of the ratio.
  - E.g., if a store sells 6 red shirts and 8 green shirts, the ratio of red to green shirts is 6 to 8.
- A **rate** is a special ratio in which the two terms are in different units.
  - E.g., if a 12-ounce can of corn costs 69¢, the rate is 69¢ for 12 ounces.

https://www.hmhco.com/blog/teaching-ratios-and-unit-rates-in-math

Summarizing Ratios

Trial	Program A	Program B	A/B
1	9	3	3.00
2	8	2	4.00
3	2	20	0.10
4	10	2	5.00
Mean	7.25	6.75	3.03

Conclusion

Program B is > 3 times better than A.

Summarizing Ratios

Trial	Program A	Program B	A/B
1	9	3	3.00
2	8	2	4.00
3	2	20	0.10
4	10	2	5.00
Mean	7.25	6.75	3.03

Conclusion

Program B is > 3 times better than A.

WRONG!

- It does not make sense to take the arithmetic mean of a bunch of ratios.
- The mean of ratios is not the same as the ratio of the means.

Turn the Comparison Upside-Down

Trial	Program A	Program B	A/B	B/A
1	9	3	3.00	0.33
2	8	2	4.00	0.25
3	2	20	0.10	10.00
4	10	2	5.00	0.20
Mean	7.25	6.75	3.03	2.70

Paradox

If we look at the ratio B/A, then A is better by a factor of almost 3.

Observation

The arithmetic mean of A/B is NOT the inverse of the arithmetic mean of B/A.

Geometric Mean

Trial	Program A	Program B	A/B	B/A
1	9	3	3.00	0.33
2	8	2	4.00	0.25
3	2	20	0.10	10.00
4	10	2	5.00	0.20
Mean	(a) 7.25	(a) 6.75	(g) 1.57	(g) 0.64

Formula

$$\left(\prod_{i=1}^{n} a_i\right)^{1/n} = \sqrt[n]{a_1 a_2 \cdots a_n}$$

Observation

The geometric mean of A/B IS the inverse of the geometric mean of B/A.

33

SPECRatio

SPECRatio normalizes the execution times to a reference computer by dividing the time on the reference computer by the time on the computer being rated, yielding a ratio proportional to performance.

Benchmarks	Sun Ultra Enterprise 2 time (seconds)	AMD A10-6800K time (seconds)	SPEC 2006Cint ratio	Intel Xeon E5-2690 time (seconds)	SPEC 2006Cint ratio	AMD/Intel times (seconds)	Intel/AMD SPEC ratios
perlbench	9770	401	24.36	261	37.43	1.54	1.54
bzip2	9650	505	19.11	422	22.87	1.20	1.20
gcc	8050	490	16.43	227	35.46	2.16	2.16
mcf	9120	249	36.63	153	59.61	1.63	1.63
gobmk	10,490	418	25.10	382	27.46	1.09	1.09
hmmer	9330	182	51.26	120	77.75	1.52	1.52
sjeng	12,100	517	23.40	383	31.59	1.35	1.35
libquantum	20,720	84	246.08	3	7295.77	29.65	29.65
h264ref	22,130	611	36.22	425	52.07	1.44	1.44
omnetpp	6250	313	19.97	153	40.85	2.05	2.05
astar	7020	303	23.17	209	33.59	1.45	1.45
xalancbmk	6900	215	32.09	98	70.41	2.19	2.19
Geometric mean			31.91		63.72	2.00	2.00

Figure 1.19 SPEC2006Cint execution times (in seconds) for the Sun Ultra 5—the reference computer of SPEC2006—and execution times and SPECRatios for the AMD A10 and Intel Xeon E5-2690. The final two columns show the ratios of execution times and SPEC ratios. This figure demonstrates the irrelevance of the reference computer in relative performance. The ratio of the execution times is identical to the ratio of the SPEC ratios, and the ratio of the geometric means (63.7231.91/20.86 = 2.00) is identical to the geometric mean of the ratios (2.00). Section 1.11 discusses libquantum, whose performance is orders of magnitude higher than the other SPEC benchmarks.

[John L. Hennessy, David A. Patterson: Computer Architecture - A Quantitative Approach, 6th Edition. Morgan Kaufmann 2017.]

Comparing Two Programs

Q. You want to know which of two programs, A and B, is faster, and you have a slightly noisy computer on which to measure their performance. What is your strategy?

Comparing Two Programs

Q. You want to know which of two programs, A and B, is faster, and you have a slightly noisy computer on which to measure their performance. What is your strategy?

A. Perform n head-to-head comparisons between A and B, and suppose A wins more frequently. Consider the null hypothesis that B beats A, and calculate the *P-value*: "If B beats A, what is the probability that we'd observe that A beats B more often than we did?" If the P-value is low, we can accept that A beats B.

(See Statistics 101.)

NOTE: With a lot of noise, we need lots of trials.

Fitting to a Model

Suppose that I have gathered this data:

Program	Time (s)	Instructions	Cache misses
python	34864	170889186565542	36615004052
java	2618	7509707536406	39322034007
C gcc -O0	1480	2274589361551	68047140354
C gcc -O3	430	278479001783	34049504541

I want to infer how long it takes to run an instruction and how long to take a cache miss.

I guess that I can model the runtime `T` as

$$T = a \cdot I + b \cdot C ,$$

where

- `I` is the number of instructions, and
- `C` is the number of cache misses.

Least-Squares Regression

A *least-squares regression* can fit the data to the model

$$T = a \cdot I + b \cdot C ,$$

yielding

- $a = 0.2002$ ns
- $b = 18.00$ ns

with $R^2 = 0.9997$, which means that 99.97% of the data is explained by the model.

Issues with Modeling

Adding more basis functions to the model improves the fit, but how do I know whether I'm overfitting?

- Removing a basis function doesn't affect the quality much.

Is the model predictive?

- Pick half the data at random.
- Use that data to find the coefficients.
- Using those coefficients, fid out how well the model predicts the other half of the data.

How can I tell whether I'm fooling myself?

- Triangulate.
- Check that different ways of measuring tell a consistent story.
- Analogously to a spreadsheet, make sure the sum of the row sums adds up to the sum of the column sums.