Performance Metric Properties

Performance Evaluation of Computer Systems

Vojtěch Horký Peter Libič Petr Tůma

Department of Distributed and Dependable Systems
Faculty of Mathematics and Physics
Charles University

2010 – 2021

Outline

Performance Metric

A performance metric is a quantitative measure of some property of interest.

Typically, they are one of:

- Count of an event of interest,
- Duration (interval between events),
- Size (value) of a parameter of interest.

We are not that much interested in formal properties.
But we still have many practical requirements.

Properties of Interest

Speed

- System completes the task successfully and provides correct results.
- We are interested in how fast it performs the task.

Efficiency

- System completes the task successfully and provides correct results.
- We are interested in how many resources were used.

Reliability

- System completes the task but the result is incorrect.
- We can measure how often the errors happen.

Availability

- The system did not perform the task because it was down.
- We can measure how much the system is (not) available.

Outline

Good Performance Metric

What is a good performance metric ?

Goals

Comparing two computer systems ?
Evaluating an optimization ?
Estimating execution cost ?

Audience

Developers ? Researchers ? Customers ? Private or public ?

Dangers

Poorly chosen metrics can be misleading !

- Hard to interpret.
- Leading to incorrect conclusions.
- Measuring features that are not interesting.

Practical Requirements for Good Metric

A good metric should be:

- Linear.
- Reliable.
- Repeatable.
- Easy to measure.
- Consistent.
- Independent.

These goals cannot always be met and can be contradictory.
But it is good to get close.

Based on Lilja: Measuring Computer Performance ... doi:10.1017/CBO9780511612398

Requirement: Linearity

Why ?

Linear metrics are easier for humans to interpret.

- If a metric doubles its value, the system should be twice as fast, or finish the task in half the time.
- Linearity is not met by many metrics:
  - Well known example is dB (acoustic pressure).
  - Also camera resolution vs image dimensions.
  - Or cache size vs miss rate or speed up.
  - They are not wrong, but may be much harder to interpret.

Requirement: Reliability

Why ?

We expect better values indicate better systems.

- One system outperforms another when the metric values indicate so.
- Many reasonable examples:
  - Network bandwidth, copying files over faster network should be faster.
  - Memory speed, running on faster memory should be faster.
  - But what about processor clock speed ?
  - Or cost ?
- Hard to guarantee for very general metrics (performance is application specific).
- Quite obvious, but often not met !

Requirement: Repeatability

Why ?

We expect the metric to be an inherent system property, hence repeatable.

- Each run of an experiment should give the same value of a metric.
- Not completely realistic:
  - Computers are not always deterministic
    (randomized algorithms, asynchronous interrupts ...).
  - Full control over experiment not always possible
    (distributed systems, database servers, cloud ...).
  - Statistical methods can help attribute variability.
  - Metric can be deterministic, thus repeatable
    (number of instructions in a program repeatable but not reliable).

Requirement: Ease of Measurement

Why ?

Obviously, if we cannot get metric values it is a problem ...

- A metric should be easy to measure
  or infer from other (easily measurable) metrics.
- More difficult to measure means more likely measured incorrectly.
  - One way network latency.
  - Single thread power consumption.
  - Timing of synchronization construct in code
    (measurement difficulty not strictly property of metric).

Requirement: Consistency

Why ?

Same meaning everywhere facilitates system comparison.

- A metric should have the same units everywhere.
- The units should have the same meaning everywhere.
- Metrics like MIPS or MFLOPS do not follow this obvious requirement.

Requirement: Independence

Why ?

Trust in metric requires system (and hence vendor) independence.

- Systems should not be optimized for particular metric.
- But vendors are known to optimise for specific benchmark (metric) !
  - For example nVidia and 3DMark, Sun and SPECjbb2000.
  - This makes evaluation results less representative.
- But...
  - Developers need benchmarks to test and optimize their code.
  - For compilers, SPEC CPU seems to be a good set, tries to be representative.

Even an initially independent metric can become an optimization target !

Outline

Selecting Metrics For an Experiment

1. List all metrics possibly measurable in the given scenario.
2. Select a reasonable subset following these criteria:
   - Low variability
     Helps to reduce number of required repetitions.
     Computing ratio usually increases variance, better avoid.
   - Non redundancy
     If one metric can be derived from another, choose only one.
   - Completeness
     Try to select so many metrics that all possible outcomes are included.
   - Insight
     Choose metrics that provide insight or validate hypotheses.
3. While executing experiments, watch for anomalies,
   extend observed metrics, then repeat.

Based on Jain: The Art of Computer Systems ... ISBN 978-0-471-50336-1