Evaluate Collaboratory

http://evaluate.inf.usi.ch/

Technical Report #1 – February 15, 2012

Can you trust your experimental results?

Stephen M. Blackburn (Australian National University), Amer Diwan (Google, University of Colorado), Matthias Hauswirth (University of Lugano), Peter F. Sweeney (IBM T. J. Watson Research Center), José Nelson Amaral (University of Alberta), Vlastimil Babka (Charles University, Prague), Walter Binder (University of Lugano), Tim Brecht (University of Waterloo), Lubomír Bulej (Charles University, Prague), Lieven Eeckhout (Ghent University), Sebastian Fischmeister (University of Waterloo), Daniel Frampton (Australian National University), Robin Garner (Southern Cross University, Lismore), Andy Georges (Ghent University), Laurie J. Hendren (McGill University), Michael Hind (IBM T. J. Watson Research Center), Antony L. Hosking (Purdue University), Richard Jones (University of Kent), Tomas Kalibera (University of Kent), Philippe Moret (University of Lugano), Nathaniel Nystrom (University of Lugano), Victor Pankratius (Karlsruhe Institute of Technology), Petr Tuma (Charles University, Prague)

ABSTRACT

Many contributions in computer science rely on quantitative experiments to validate their efficacy. Well-designed experiments provide useful insights while poorly-designed experiments can mislead. Unfortunately, experiments are difficult to design and even seasoned experimenters can make mistakes. This paper presents a framework that enables us to talk and reason about experimental evaluation. As such, we hope it will help our community to avoid and recognize mistakes in our experiments. This paper is the outcome of the Evaluate 2011 workshop whose goal was to improve experimental methodology in computer science.

Introduction

Quantitative experiments are central to scientific investigation. A well-designed experiment adds to scientific knowledge. In contrast, a poorly-designed experiment can mislead scientists because it does not support the experimenter's claim (e.g., "A is better than B"). Unfortunately, even with our best diligence, we may make a mistake and end up with a poorly-designed experiment.

This paper describes a framework for understanding and reasoning about many of the common mistakes in experiment design that we have observed in the computer science literature. Our hope is that this framework will: a) expose common concepts that foster effective communication about experimentation, b) help the reader understand and avoid mistakes in their own experiments, and c) help reviewers reason with clarity about the strengths and shortcomings of papers submitted for publication. We hope that together these outcomes will serve to improve the quality of experimental evaluation in computer science.

There are many books that provide a recipe for constructing good evaluations[1,2,3]. This paper does not try to duplicate them. Instead it complements them by providing a framework for understanding and critiquing evaluations.

Experimental Evaluation

The framework in this paper applies to any quantitative experimental evaluation. It applies to researchers as well as practitioners, and it is independent of the particular scientific process they use. The framework only assumes that experimenters perform an experiment, and that they make a claim based on the results.

Components of an Experiment

When designing quantitative experiments, an experimenter must consider the following components (Figure 1): (i) **measurement contexts**, which identify the software and hardware components to vary or hold constant in the experiment; (ii) **workloads**, which identify the benchmarks, along with their inputs, to use in the experiment; (iii) **metrics**, which identify the properties to measure and how to measure them; and (iv) **data analysis and interpretation**, which identifies how to analyze the data and how to interpret the results of the analysis to provide insight into resulting claims.

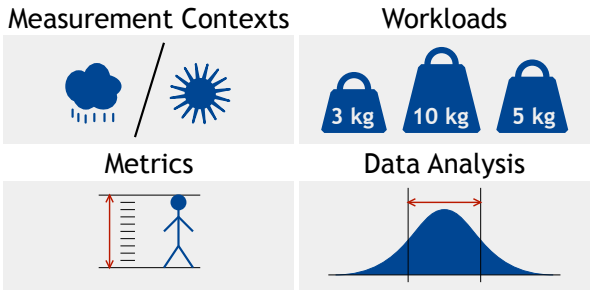

Figure 1. Components of an Experiment

The measurement context and workload together define the *control* and *independent* variables for an experiment. They represent the *knobs* the experimenter holds constant (control) or changes (independent). For example, an experiment to evaluate a compiler optimization may perform measurements with different optimization levels (i.e., the independent variables) while keeping the same hardware and operating system across the experiments (i.e., the control variables).

The metrics and data analyses together define the *dependent* variables of an experiment. They represent the *gauges* that the experimenter reads. For example, the experimenter may determine the mean and confidence interval for the measured execution time.

While related, measurement context and workloads are different components of an experiment. Intuitively, the measurement context is the environment in which the experimenter places the measured system and the workload is the load that the experimenter applies to the measured system.

A poorly-designed experiment makes poor choices in one or more components.

Pitfalls in Experimentation

For each of these four components, this paper identifies four common pitfalls (Figure 2): inappropriate, ignored, inconsistent, and irreproducible. We abbreviate these as the four '*I*'s. Each pitfall represents a poor choice for a component. An **inappropriate** component includes elements that are inappropriate for the experimenter's claim; for example, including a desktop workload when making claims about a supercomputer. An **ignored** component omits aspects relevant to the claim; for example, ignoring compile time when making claims about the quality of a just-in-time compiler. An **inconsistent** component compares aspects that are inconsistent with each other; for example, making claims about the benefit of ***A*** compared to ***B*** while comparing ***A*** on a modern system to ***B*** on an antiquated system. An **irreproducible** component is one that others cannot use to reproduce the experiments; for example an inadequately described data analysis technique hinders reproducibility.

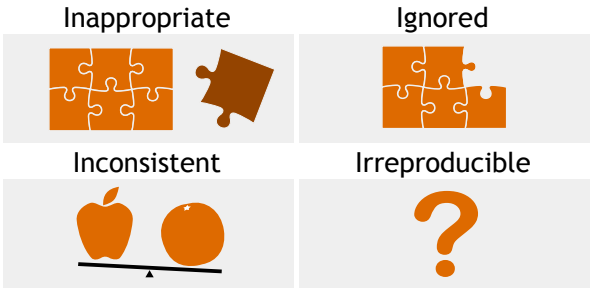

Figure 2. Pitfalls in Experimentation

The Scope of the Claim and the Evaluation

It is difficult to avoid these pitfalls when the claim is broad. On the other hand, a narrow claim may be of limited usefulness. For example, it is much harder to design an experiment to support the claim "optimization ***O*** reduces energy consumption of all mobile applications" than it is to support the claim "optimization ***O*** reduces the energy consumption of a *specific* movie player displaying a *specific* movie at a *specific* resolution on a *specific* mobile phone".

The relationship between the scope of the evaluation and the scope of the claim is key. The more general the experimenter's claim, the more difficult it will be for them to provide a sufficiently broad evaluation. On the other hand, the narrower the claim, the less interesting and meaningful it is likely to be. Understanding this relationship between the scope of the evaluation and the scope implied by the claim is an optimization problem that lies at the heart of experimental design.

The remainder of this paper develops this framework, giving concrete and realistic examples of the four pitfalls for each of the four components of an experiment design.

Measurement Contexts

The measurement context defines the environment in which an experiment runs. For example, when evaluating the performance of a middleware platform, the measurement context includes the specific hardware and operating system versions.

Scenario: To illustrate the four pitfalls that may occur for a specific measurement context, assume that you are an engineer at a smartphone manufacturer.

Inappropriate Measurement Contexts

A measurement context is inappropriate when it is flawed or does not reflect the measurement context that is implicit in the claim. This may become manifest as an error or as a distraction (a "red herring").

Example: You **claim** that your smartphone battery lasts 10 hours when running your newspaper application. Your **evaluation** ran the application while the phone was in airplane mode, which caused your application to use data cached locally on the phone instead of fetching data from a server over the wireless network.

Ignored Measurement Contexts

An aspect of the measurement context is ignored when an experiment design does not consider it even when it is necessary to support the claim.

Example: You **claim** that your phone drops fewer than 1% of calls. Your **evaluation** ignored left-handed users, who, because they hold the phone differently, may interfere with the antenna.

Inconsistent Measurement Contexts

A measurement context is inconsistent when an experiment compares two systems and uses different measurement contexts for each system. The different contexts may produce incomparable results for the two systems. Unfortunately, the more disparate the objects of comparison, the more difficult it is to ensure consistent measurement contexts. Even a benign-looking difference in contexts can introduce bias and make measurement results incomparable. For this reason, it may be important to randomize experimental parameters (e.g., memory layout in experiments that measure performance).

Example: You **claim** that the optimizations that you implemented in the most recent version of your multi-threaded mobile phone application led to a 20% reduction in response time. Your **evaluation** compared the response time with optimizations on a dual-core phone against the response time without optimizations on a single-core phone.

Irreproducible Measurement Contexts

If the measurement context is irreproducible then the experiment is also irreproducible. Measurement contexts may be irreproducible because either they are not public or they are not documented.

Example: You **claim** that your phone's operating system boots in less than 5 seconds. Your **evaluation** fails to report which smartphone model you used in your experiment to obtain those results.

Workloads

Experimental evaluation typically involves subjecting the system, or systems, to one or more workloads. A workload may be characterized as a benchmark with a given set of inputs.

Scenario: To illustrate the four pitfalls that may occur for workloads, assume that you are responsible for the high-frequency trading system of a large financial institution.

Inappropriate Workloads

A workload is inappropriate when it is flawed or does not reflect the workload that is implicit in the claim.

Example: You **claim** that your algorithmic trading system completes 99.9% of transactions within 3 milliseconds. Your **evaluation** used a workload that represents quiescent markets, which does not capture high-frequency trading activity.

Ignored Workloads

In the real world, most systems are subjected to diverse workloads. The selection of workloads must include sufficient diversity to support the intended claims. Alternatively, if the workloads are not sufficiently diverse, the claim must be narrowed appropriately.

Example: You **claim** that your newly developed high-frequency trading algorithm outperforms the currently used algorithm. Your **evaluation** used a workload that ignores one of the largest markets in which your institution participates and on which your system is employed.

Inconsistent Workloads

When an evaluation compares two or more systems, it is essential that the systems under test are subjected to the same workloads.

Example: You **claim** that your new system consumes only 10% more power than your old system. Your **evaluation** compared the old system's power consumption as measured during the "Flash Crash" of May 6, 2010, to the new system's consumption during a day when markets were stable.

Irreproducible Workloads

The workloads used to evaluate an innovation should include at least some workloads that others have access to; this will enable others to reproduce the results. We are careful not to say: "one should only pick workloads that others can access" because sometimes valuable insight can be gained from commercially pervasive, but proprietary, workloads that cannot be disclosed.

Example: You **claim** that an upgrade of the operating system will reduce the median trading latency by up to 10%. Your **evaluation** used a workload consisting of real-time market data, which you fed concurrently to both systems. However, you neither captured nor characterized that workload in any way, making it impossible for someone else to examine a similar workload.

Metrics

In quantitative empirical studies, metrics are necessary to quantify certain properties of systems. Specifically, metrics identify the properties that the experiment will measure and how those properties will be measured. Metrics can range from the end-to-end execution time of a system, to the size of a data structure, to the precision of static analysis results.

Scenario: To illustrate how the four pitfalls apply to metrics, assume you are a researcher at a supercomputing center working on compiler optimizations to improve supercomputer performance.

Inappropriate Metrics

A metric is inappropriate when it is flawed or when it does not support the intended claims of the experiment. A common manifestation of an inappropriate metric is the use of a *surrogate* metric, which is easy to measure, but which may not correlate with the desired metric it replaces.

Example: You **claim** that your new compiler optimization speeds up programs. Your **evaluation** used instructions per cycle (IPC) to measure performance with and without your optimization. Because your optimization may change the number of instructions, IPC is inappropriate. Your optimization might increase IPC by inserting extra no-op instructions that execute quickly and increase IPC without improving overall performance.

Ignored Metrics

A metric is ignored when it is excluded despite being necessary to confirm an

evaluation's claims. There are two primary manifestations of "ignored metrics".

First is to ignore the cost of an innovation while reporting only the benefit. Most innovations have both cost and benefit and ignoring the cost is misleading.

Second is to report only ends-based or only means-based metrics. Both are important: ends-based metrics (such as end-to-end execution time) are often relevant to the claim; means-based metrics (such as cache misses) are necessary for confirming the *cause* of the change in the ends-based metric. Thus reporting just ends-based or just means-based metrics is misleading.

Example: You **claim** that your new data-locality optimization improves performance. Your **evaluation** only measured execution time (an ends-based metric), but did not measure the improvement in data locality, e.g., by counting data cache misses (a means-based metric). It thus failed to show that the improved performance was caused by an improvement in data locality.

Inconsistent Metrics

To demonstrate the superiority of an innovation, many evaluations compare that innovation to an existing baseline. If one metric is used for the baseline, and a (even slightly) different metric is used for the innovation, the metrics are inconsistent and the measurement results are not necessarily comparable.

Example: You **claim** that your optimization reduces L2 cache misses. Your **evaluation** of the unoptimized program includes both demand-load and prefetch request misses, but the optimized version only counts demand-load misses.

Irreproducible Metrics

A metric's name may seem to unambiguously define the meaning of that metric, however, often the actual implementation of the metric leaves a lot of flexibility. A study needs to precisely define how the metric is measured, otherwise future studies cannot produce comparable measurements.

Example: You **claim** that your compiler optimization reduces the number of executed instructions. Your **evaluation** failed to report whether you included the number of speculatively executed instructions in your count.

Data Analysis and Interpretation

Data analysis takes as input the measured metric values (data) generated by running workloads in a measurement context. This component determines how to analyze the data and interpret the results of the analysis.

Scenario: To illustrate the four pitfalls that may occur for data analysis and interpretation, assume that you are a software engineer developing new functionality for your company's web site to support a new sales promotion. The new promotion is expected to quadruple traffic to the company's web site; however, the new functionality must not significantly degrade user response time.

Inappropriate Data Analysis

A common goal of data analysis is to draw conclusions about a population from a small subset of the population (a sample). For this generalization to be valid, the experiment must use appropriate analysis methods.

Example: You **claim** that your sales-promotion feature does not lead to perceptibly longer response times. Your **evaluation** took a sample of 50 measurements and computed the minimum response time. The minimum is not

an appropriate statistic because the minimum may be a low-probability outlier.

Ignored Data Analysis

Well-studied methods for analyzing data are used widely among the experimental sciences. Using these methods reduces the risk of poor data analysis and interpretation.

Example: You **claim** that the mean response time does not significantly increase. Your **evaluation** took a sample of 50 measurements and computed the mean. However, you ignored any inferential statistics, such as confidence intervals, that would help to establish the statistical significance of your result.

Inconsistent Data Analysis

Inconsistent data analysis occurs when different data analysis techniques are applied to data pertaining to different objects of an experimental study.

Example: You **claim** that your new feature does not increase the response time. Your **evaluation** used the mean to compute the response time of the new system. You compared that mean to the median response time you found in a report about the performance of the prior system.

Irreproducible Data Analysis

For a data analysis to be reproducible, the experiment must carefully document the analysis it uses and identify and report any biases in the analysis. If an experimenter does not document their analysis and explicitly state their biases, other researchers have little hope of reproducing the analyses and arriving at the same conclusion.

Example: You **claim** that the increase in response time is not statistically significant. Your **evaluation** did not report the confidence level, the corresponding confidence interval, and the number of samples taken.

Conclusions

Although many of us can already recognize poorly-designed experiments (at least in our own areas), we do not have a common vocabulary for clearly and succinctly explaining the shortcomings of an experiment. This paper is an attempt to provide such a vocabulary. It is an outcome of the Evaluate 2011 workshop, which brought together the authors of this paper, who are concerned with the poor state of evaluation in their areas.

No experimental evaluation is perfect. For example, most experiments use a set of benchmarks that sample a range of workloads. Using a sample is acceptable as long as the sample is "representative". We believe this framework will help to identify both the strengths and shortcomings of an experimental evaluation. As a reader, you must balance the shortcomings with the strengths to determine if the experimental evaluation is acceptable. As a practitioner, understanding the shortcomings can help you to improve your experiment's design, modify your claims, or, when appropriate, argue why a shortcoming is acceptable.

We hope that this framework enables you to perform better quantitative experiments and to identify the shortcomings and strengths of others' experiments with clarity.

---

[1] Raj Jain, "The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling," John Wiley & Sons, New York, NY 10158-0012, ISBN 0471-50336-3, 720 pp. 1991.

[2] David Lilja, “Measuring Computer Performance: A Practitioner's Guide,” Cambridge University Press, 280 pp. 2005.
[3] Sam Kash Kachigan, “Statistical Analysis: An Interdisciplinary Introduction to Univariate & Multivariate Methods”, Radius Press, New York, NY, ISBN-10: 0942154991, 589 pp. 1986.