Repeatability & Workability for the Software Community: challenges, experiences, and the future

D. Shasha[1]
[1] (shasha@cs.nyu.edu) New York University, New York, USA

ABSTRACT

It's much easier to test claims in computer science than in natural science. So, we[1] decided to give it a try in the database conference ACM SIGMOD 2008. Specifically, the repeatability committee volunteered to assess the results of database experimental papers [1] if authors chose to submit their code and data for the purpose. For SIGMOD 2009, the testing expanded to "workability" – varying data and parameters to see how the software behaves under conditions that are different from the ones the researchers reported. This short note describes what we have learned over the past two years, the challenges we have faced, ones we still face, and directions we suggest for the future.

1. THE GOAL

In natural science, repeatability allows one scientist to verify the assertions of another, occasionally exposing fraud, but more often simply providing a check against exhuberant claims. Natural science papers conform to the repeatability requirement by providing a complete description of the protocol used in an experiment (reagents, equipment used down to the model number, times, temperatures etc.). The protocol must be described in sufficient detail for another lab to replicate the experiment. The accuracy of repeatabililty is far from perfect. The biologists I work with often talk about "biological variation" – the differences in results that come about from the vagaries of living organisms or (more embarrasingly) different laboratories.

Computer science papers can't describe protocols in enough detail for repeatability because software is far more complex than laboratory procedures. Fortunately for computer science, however, computational researchers can describe the core of their algorithms in a paper and then provide software and data to enable repeatability on another researcher's computer or cluster. Also, fortunately for computer science, it is easy to change data in order to test "workability": does the software behave well and predictably for slight changes in the data? Finally, and again fortunately for computer science, preparing code and data for repeatability/workability leads, without much additional work, to preparing the code for archiving and distribution, thus greatly enhancing the usefulness of the work done.

Almost everyone sees virtues in these goals. Shamelessly taking advantage of my role as program chair, I requested a hand vote in the opening session of SIGMOD 2008 on the question: "If you had time, would you participate in a repeatability effort on any system you built?" The vote in favor of repeatability was overwhelming, something like 100 (we didn't count) to 2.

There are however some challenges.

1. Intellectual property rights may prevent some researchers from submitting code and/or data. For this reason, repeatability or workability in the database community has been and will continue to be voluntary.

2. Preparing code and data for repeatability testing entails a lot of work on the part of authors. First it requires documentation, the explicit rendering of all dependencies on compilers, operating systems, and hardware. Second, it requires incentives. In SIGMOD 2008, the incentive was a sentence authors could add to the copy of their article in the proceedings: "The results in this paper were verified by the SIGMOD repeatability committee." That was sufficient to get 2/3 of the writers of accepted papers to attempt repeatability. In SIGMOD 2009, validated authors were invited to set up a Wiki on an ACM site. Academics are motivated by reputation – designing repeatable software enhances one's reputation. We have to get the word out.

3. Assessing code and data for repeatability and workability requires a lot of effort on the part of the testers. Platforms differ. Computation time can be significant (e.g. for parallel code). Documentation is rarely perfect. Participating in a repeatability committee should count as an important community service.

2. EXPERIENCES

When we first announced the repeatability initiative in the SIGMOD 2008 webpage, we received some revealing excuses:

We cannot distribute code and data because the author has moved, making the retrieval of code and data infeasible at this point.

The subsets were chosen randomly from a large dataset, and unfortunately no trace about the identity of the used documents has been kept.

[1]Ioana Manolescu and Stefan Manegold have led the effort over the past three years and have recruited outstanding system researchers to help out

We received various positive comments:

This wasn't too hard and I think it was definitely worth it. We even found a mistake (thankfully a minor one, not affecting our conclusions)...

Overall, of the 78 accepted papers, 53 attempted repeatability (and 29 achieved it). After the process was over, we sent a survey asking about experiences. Here were some comments.

Some supported our main goal. *I think this is a noble effort and costs almost nothing for authors if they set up experiments with repeatability in mind. The focus on repeatability will lead to better science in our community.*

Others suggested that the "beneficial side effect" of archiving was even more important. *Sharing experiments (code and data) benefits a lot of researchers, especially small groups.*

Some just thought repeatability required too much work. *I think experimental repeatability is an important thing, and I support the motivation. But I believe that the current mechanism is just plain wrong. Way too much work for the benefit derived.*

Unfortunately, the author did not suggest an alternative mechanism, but since you might hear a similar objection if you try this, here is at least part of the argument you might use. The simple fact is that computer science is an engineering discipline not simply a mathematical one. Depending on your subspecialty, you know that there is an enormous variety of multidimensional data structures, machine learning algorithms, and linear programming methods. Good engineering or the right combination of engineering and problem can determine the best choice of data structure or algorithm. That is why experiments are necessary and why workability tests can help to show the reach of an implementation. Particularly in those systems fields where heuristics are necessary, experiments and their variants are critical.

3. RECOMMENDED REPEATABILITY PROTOCOL FOR THE COMPUTATIONAL SCIENCES

1. Announce the specifications of the voluntary repeatability and workability effort along with the call for papers. As the comments above reveal, researchers are not always careful to record all the parameters they set to get their results.

2. Test only accepted papers. This has two benefits: (i) it reduces the workload for the testers and (ii) authors seem willing to put in the extra work if their paper has been accepted. The disadvantage is that waiting till papers are accepted before performing the evaluation makes it difficult to finish the evaluation before the camera ready deadline, so the opportunity to trumpet the success of the evaluation has to wait till after the conference. In SIGMOD 2009, the results were available at the time of the conference, so authors were able to announce their successes during their talks.

3. Ask authors to include in their submission information the length of time their experiments are expected to run. Also, ask them to give guidelines about how to extend their experiments beyond the contents of their paper. Possibilities range from explanations of how to use different data sets, query work loads, tuning and/or configuration parameters to compilation, and installation instructions for alternative hardware/software environments.

4. Assign papers to reviewers based on the compatibility of the papers' hardware and software requirements with the reviewers' resources, while avoiding conflicts of interest. Each paper should be assigned two reviewers: a *primary* reviewer to do the actual repeatability and workability evaluation, and a *secondary* reviewer as back-up and to double-check the primary reviewers report.

5. Adopt an anonymous web-based communication channel between reviewers and authors so reviewers can ask questions. This is needed because successful repeatability can be hindered or even prevented by minor problems in setting up and running the experiments due to missing details in the provided instructions.

6. Make sure the evaluation results are visible. The SIGMOD PubZone server [2] is an example venue for this purpose.

4. RECOMMENDED TOOLS

Imagine that you have had your paper accepted and you would like to ready your software and data for repeatability and workability testing. What kind of tools would you need? Here is a partial list:

1. A utility that runs through the source code detecting machine dependencies (e.g. machine language), operating system dependencies (e.g. system calls), shell dependencies (e.g. shell scripts), path dependencies (e.g. absolute path names), compiler dependencies (e.g. language constructs), and browser dependencies. These should be identified and either corrected or noted as constraints on the tester. As a start, a CMakeLists file for all directories and two or more tested build environments will help ensure portability. A git repository may help distributing the software and proposed workability changes.

2. A computational resource where code that has been purged of dependencies can be tested by the author. That resource should include a variety of platforms perhaps as virtual machines.

3. A testing harness for workability that will allow the execution and collection of results of software using different parameters. This will be useful for both the author and the tester.

5. SUMMARY

The computational community can be justly proud of its practical impact over the years. This is due to a confluence of good ideas with good engineering. Repeatability and Workability help to ensure the validity of good ideas and provide paradigms and platforms for good engineering. Systems researchers understand the value of the effort. It's the right thing to do.

6. ACKNOWLEDGMENTS

Ioana Manolescu and Stefan Manegold have realized the repeatability vision through their leadership. The excellent team they have recruited have made it a reality. Brad Barber made some very helpful suggestions to this manuscript.

7. REFERENCES

[1] I. Manolescu, L. Afanasiev, A. Arion, J.-P. Dittrich, S. Manegold, N. Polyzotis, K. Schnaitter, P. Senellart, S. Zoupanos, and D. Shasha. The repeatability experiment of SIGMOD 2008. *SIGMOD Record*, 37(1):39–45, Mar. 2008.

[2] PubZone: scientific publication discussion forum. `http://www.pubzone.org`.