Draft Repeatability Evaluation Proposal
Hybrid Systems Computation & Control

Ian M. Mitchell
Department of Computer Science
University of British Columbia
mitchell@cs.ubc.ca

November 4, 2013

Abstract

We propose to allow authors of accepted papers at HSCC to submit a repeatability package (whose contents are outlined below) around the same time as the final version of the paper. That package will be reviewed by members of a repeatability evaluation committee according to the criteria described in this document. Those papers corresponding to packages which pass the criteria will be identified as "repeatable" at HSCC and possibly in the ACM DL.

1 Introduction

Modern engineering and science depend heavily on computational results, but rarely are those computations peer reviewed to the same level as theoretical or experimental results. Occasionally it leads to a very public disaster [7, 9], but more often it simply slows and frustrates the day-to-day research activities of everybody, including those who did the research in the first place.

This proposal outlines an optional repeatability evaluation procedure for papers accepted to HSCC with two goals in mind:

- To improve the reproducibility of computational results from HSCC and thereby raise the profile of HSCC publications and authors. Although there has been no direct study of the benefits of sharing code, there is evidence that sharing research data improves citation counts in the biosciences [10]. Since at least 2011 the majority of HSCC papers have had a significant computational component so there is likely to be a large pool of suitable research from which to draw. HSCC also has an opportunity to distinguish itself in this respect, since to our knowledge such a process has not been used in the formal methods or cyber-physical systems domains, although it has been used in a number of conferences in computer science (see section 2.

- To provide authors an opportunity and incentive to adopt best-practices that are known to improve the quality of compuational results. Despite the prevalence of computational techniques in modern research, few scientists or engineers receive formal training in either domain [11, 8, 5]. Articles are available which collect best-practices; for example, [13, 1, 4, 6, 12]; however, reading about best-practices and actually instituting them in a busy research lab are two different things. By providing a concrete, short-term benefit in the form of additional acknowledgement at HSCC as well as clear guidelines of how to achieve that goal, the repeatability evaluation procedure will encourage researchers to adopt better code and data management procedures. After the first year there will also be a collection of domain-appropriate examples which model repeatable outcomes for new participants.

We seek to achieve these benefits without disturbing the current process through which HSCC has generated high-quality programs for the past sixteen years. Therefore, the current submission, review and acceptance procedures, as run by the program committee and managed by the PC chairs, will continue as usual. Only once acceptance decisions are finalized will the authors of accepted papers be invited to take part in this new repeatability evaluation process. The repeatability evaluation process will have no effect on whether a paper is accepted to HSCC and will be entirely optional. Only papers that are successful in the repeatability evaluation process will be identified as such, while those which do not submit or are unsuccessful will just appear as usual in the proceedings and ACM Digital Library.

2 Other Repeatability and Artifact Submission Efforts

A number of conferences have recently adopted similar processes, and we summarize their experience here. The first such conference appears to have been SIGMOD (a large ACM conference on database research) which has been running an optional "experimental reproducibility effort" since 2008 [3]. More recently several conferences in software engineering have adopted "artifact evaluation" processes, including ESEC/FSE 2011 & 2013, SAS 2013, ECOOP 2013 & 2014 and OOPSLA 2013 [1]. The co-chairs of OOPSLA 2013 have produced an "Artifact Evaluation Artifact" [2] which provides a detailed description of their process including timeline, modifications to the standard submission process, announcements and emails sent to the community and to the committees, etc.

A number of common elements emerge from these efforts which we are choosing to emulate:

- Evaluation of the artifacts / reproducibility is independent of the acceptance decision for the related paper, only accepted papers are invited to take part in the extra

[1] See `http://cs.brown.edu/~sk/Memos/Conference-Artifact-Evaluation/`, `http://esec-fse.inf.ethz.ch/program_artifacts.html`, `http://research.microsoft.com/en-us/events/sas2013/`, `http://ecoop13-aec.cs.brown.edu/`, `http://ecoop14.it.uu.se/calls/artifacts.php` and `http://splashcon.org/2013/cfp/due-june-01-2013/665-oopsla-artifacts`

evaluation step, and the extra evaluation step is optional. This choice is to avoid upsetting an already well established and presumably effective process for choosing the papers which will be part of the conference program.

- With the exception of the chair(s), the evaluation committee is formed by junior members of the community: senior graduate students and postdocs. As explained by Shriram Krishnamurthi in his summary of the ESEC/FSE 2011 effort, this population is likely to be more familiar with current software and to have more time to conduct experiments.

- The submitted material is treated as confidential and there is no requirement to publicly release it. This choice is made to address concerns that authors with proprietary material might be discouraged from submitting anything to the conference if they could not submit to the reproducibility evaluation. The hope is that once they have packaged the material for submission, the majority of authors will have little incentive not to publicly release it.

- The review process is single-blind. This approach is already standard practice in most conference venues, and the fact that the evaluation committee is made up of junior members of the community only reinforces its importance.

- The review process is not competitive, meaning that a decision as to whether a given submission satisfies the threshold of reproducibility (or whatever threshold was used for a given conference) is made independently of the decisions on other papers. This choice is in contrast to the competitive paper acceptance process used in most conferences, which typically have a limited number of slots in the program. Because only papers that have already been accepted are considered for this additional evaluation, it seems unnecessary to institute a further layer of competition and we thereby reduce the need for committee-wide discussion of the relative merits of different submissions.

3 Proposed Procedure

A repeatability evaluation committee (REC) will be formed, primarily from postdocs and senior graduate students in the HSCC community. Past and present program committee members will be invited to suggest suitable candidates, so the REC will be formed in the late summer / early fall after the PC is confirmed. The REC will operate after the final submission deadline (which is in late January / early February), so there will be no conflict with any secondary reviewing duties during the main HSCC reviewing period (typically November) or with paper preparation (September / October and January).

Authors of accepted papers will be invited in the acceptance email to submit a repeatability package (RP). The deadline for the RP will be roughly the same date as the final paper submission. A link from the HSCC web page will explain the format and criteria for the RP. Submitted RPs will be distributed to the REC for evaluation according to the rubric provided below. At least two REC members will evaluate each RP, with at least one more

in case of a discrepancy in scores. It is not expected that an extensive discussion phase will be required, since the evaluation process is based on a threshold rather than a competition.

Papers whose repeatability is confirmed would be identified in an REC report. Authors would be encouraged but not required to make their RPs publicly available.

Still to be determined:

- Should there be an authors' reply / revision phase?
- Should the RP and/or the instructions be included as supplemental material in the ACM DL?
- What components of the review should be shared with the authors and/or public?
- Is it possible to mark papers as "repeatable" in the ACM DL?

4 Sample Wording

Introducing new procedures into a smoothly functioning organization can be delicate. It is expected that there will be kinks to smooth out, so we would like to attract a reasonable but not overwhelming number of submissions in the first year; consequently, we propose a soft launch in the sense that advertisement of the repeatability evaluation option will be clear but not highlighted. With that in mind, we list some draft text for the various platforms through which HSCC communicates with the community.

- **Call for Papers:** Authors of accepted papers with a computational component will be invited to submit their code and data to an **optional** repeatability evaluation in late January. More details can be found at [website].
- **Initial Submissions Website:** The review and acceptance procedures for papers are unchanged from last year; however, authors of accepted papers with a computational component will be invited to submit their code and data to a repeatability evaluation process at roughly the same time as the final paper submissions are due. *The repeatability evaluation is entirely optional, and all accepted papers will be published as usual in the conference proceedings whether or not they take part in this process.* More details can be found at [website], and authors who may be interested in this option are encouraged to take the criteria into account as they develop the code and data supporting their initial submission.
- **Acceptance Emails:** If you feel that your paper contains a computational component, you are invited to submit a repeatability package (RP) to the repeatability evaluation committee by [as specific a deadline as possible, given the uncertainty that often surrounds the final submission deadline]. Such a submission is entirely optional, and will not effect the publication of the final version of your paper in the conference proceedings. Papers whose RPs satisfy the evaluation criteria will be highlighted as such at the conference and on the website. More details on the RPs and the evaluation criteria can be found at [website].

- **Final Submissions Website:** Authors of accepted papers interested in submitting to the repeatability evaluation process must submit a repeatability package (RP) by [deadline] to [website]. [More instructions, including links to a list of acceptable formats, a checklist of contents, and the repeatability evaluation criteria.]

In addition to these communications, we propose that:

- A brief REC chair's report appear in the proceedings to explain the repeatability evaluation process.
- A final REC report with a list of the papers judged to be repeatable be disseminated on the website, at the conference and possibly published if a suitable venue can be found.

5 The Repeatability Package

All repeatability packages (RPs) must include two components. The first is a document (preferably plain text, pdf or html) explaining (at a minimum):

- What elements of the paper are included in the RP (eg: specific figures, tables, etc.).
- The system requirements (eg: OS, compilers, environments, etc.) for running the package.
- Instructions for installing and running the software and extracting the corresponding results.

The second is the software and any accompanying data. We propose to accept at least the following formats:

- A link to a public online repository, such as `bitbucket.org`, `code.google.com`, `github.com`, or `sourceforge.net`.
- An archive file (eg: zip, bz2, tgz) containing all the necessary components.
- A virtual machine image using either VirtualBox or VMWare.

The SIGMOD conference reports that the most common reason for repeatability failure is for the reviewers to have trouble with installation [3], so authors who submit RPs in the first two forms should be particularly careful about capturing all dependencies in their RP and/or clearly documenting an installation process.

If we can recruit somebody with cloud experience, we could also possibly accept cloud instances such as Amazon's EC2, depending on whether it is possible to blind reviews.

6 Repeatability Evaluation Criteria

For each RP, each member of the REC assigned to review that RP will judge it on based on the following rubric. Each criterion is judged on the following scale:

- significantly exceeds expectations (4),
- exceeds expectations (3),
- meets expectations (2),
- falls below expectations (2),
- missing or significantly falls below expectations (0).

It is expected that very few RPs will score a 4 in any criterion. In order to be judged "repeatable" an RP must achieve an average score of at least 2 and must not have any scores of 0. Keep in mind when reading the rubric that evaluation is a threshold process, not a competition, so each RP is evaluated independently according to the objective criteria. While the higher scores in the criteria may require significant effort to achieve, a paper will be judged repeatable if it meets expectations by satisfying the following properties: (a) at least half of the computational elements in the paper are repeatable; (b) for every repeatable element there is an instruction explaining how to repeat it and the environment under which the software was originally run is described; and (c) almost all software components have clear names and/or documentation of their purpose, and the format of data files is explained. The higher scores in the criteria should be considered aspirational goals, not requirements for acceptance.

Based on this rubric, it is suggested that repeatability reviews adhere to the following template:

- Summary of the contributions of the paper.
- List of elements in the paper which were deemed to be computational and hence considered during review.
- Platform on which the RP was evaluated.
- Comments on any issues encountered during installation / startup.
- Evaluation and comments on each of the reproducibility criteria.

6.1 Coverage

What fraction of the appropriate figures and tables are reproduced by the RP? Note that some figures and tables should not be included in this calculation; for example, figures generated in a drawing program, or tables listing only parameter values. The focus is on those figures or tables in the paper containing computationally generated or processed experimental evidence to support the claims of the paper.

Note that satisfying this criterion does *not* require that the corresponding figures or tables be recreated in exactly the same format as appears in the paper, merely that the data underlying those figures or tables be generated in a recognizable format.

A *repeatable* element is one for which the computation can be rerun by following the instructions in the RP in a suitably equipped environment. An *extensible* element is one for which variations of the original computation can be run by modifying elements of the code and/or data. Consequently, necessary conditions for extensibility include that the modifiable elements be identified in the instructions or documentation, and that all source code must be available and/or involve calls to commonly available and trusted software (eg: Windows, Linux, C or Python standard libraries, Matlab, etc.).

The categories for this criterion are:

- None (missing / 0): There are no repeatable elements. This case automatically applies to papers which do not submit a RP or papers which contain no computational elements.
- Some (falls below expectations / 1): There is at least one repeatable element.
- Most (meets expectations / 2): The majority (at least half) of the elements are repeatable.
- All repeatable or most extensible (exceeds expectations / 3): All elements are repeatable or most are repeatable and easily modified. Note that if there is only one computational element and it is repeatable, then this score should be awarded.
- All extensible (significantly exceeds expectations / 4): All elements are repeatable and easily modified.

6.2 Instructions

This criterion is focused on the instructions which will allow another user to recreate the computational results from the paper.

- None (missing / 0): No instructions were included in the RP.
- Rudimentary (falls below expectations / 1): The instructions specify a script or command to run, but little else.
- Complete (meets expectations / 2): For every computational element that is repeatable, there is a specific instruction which explains how to repeat it. The environment under which the software was originally run is described.
- Comprehensive (exceeds expectations / 3): For every computational element that is repeatable there is a single command which recreates that element almost exactly as it appears in the published paper (eg: file format, fonts, line styles, etc. might not be the same, but the content of the element is the same). In addition to identifying the specific environment under which the software was originally run, a broader class of environments is identified under which it could run.
- Outstanding (significantly exceeds expectations / 4): In addition to the criteria for a comprehensive set of instructions, explanations are provided of: (a) all the major components / modules in the software, (b) important design decisions made during implementation, (c) how to modify / extend the software, and/or (d) what environments / modifications would break the software.

6.3 Quality

This criterion explores the documentation and trustworthiness of the software and its results. While a set of scripts which exactly recreate, for example, the figures from the paper certainly aid in repeatability, without well-documented code it is hard to understand how the data in that figure were processed, without well-documented data it is hard to determine whether it can be trusted, and without testing it is hard to determine whether the results are correct.

If there are tests in the RP which are not included in the paper, they should at least be mentioned in the instructions document. Documentation of test details can be put into the instructions document or into a separate document in the RP.

The categories for this criterion are:

- None (missing / 0): There is no evidence of documentation or testing.
- Rudimentary documentation (falls below expectations / 1): The purpose of almost all files is documented (preferably within the file, but otherwise in the instructions or a separate readme file).
- Comprehensive documentation (meets expectations / 2): The purpose of almost all files is documented. Within source code files, almost all classes, methods, attributes and variables are given lengthy clear names and/or documentation of their purpose. Within data files, the format and structure of the data is documented; for example, in comma separated value (csv) files there is a header row and/or comments explaining the contents of each column.
- Comprehensive documentation and rudimentary testing (exceeds expectations / 3): In addition to the criteria for comprehensive documentation, there are identified test cases with known solutions which can be run to validate at least some components of the code.
- Comprehensive documentation and testing (significantly exceeds expectations / 4): In addition to the criteria for comprehensive documentation, there are clearly identified unit tests (preferably run with a unit test framework) which exercise a significant fraction of the smaller components of the code (individual functions and classes) and system level tests which exercise a significant fraction of the full package. Unit tests are typically self-documenting, but the system level tests will require documentation of at least the source of the known solution.

Note that tests are a form of documentation, so it is not really possible to have testing without documentation. While it may appear unfair to require a uniformly high level of documentation and testing to achieve a certain category in this criterion no matter what the size of the RP, it is precisely those RP which are largest that should have more documentation and testing to demonstrate high quality. For a comparison, consider a theoretical paper. If it has a complex theorem whose long proof is broken into many lemmas, should we accept any less rigor in the proofs of any of those lemmas than we would from a paper with only a single, simple theorem?

7 Discussion

The decision to focus on only the accepted papers is driven by three factors: the experience of the conferences which have already adopted similar procedures, a reduction in workload, and the desire to avoid interfering with HSCC's well-proven review and acceptance procedure. For the same reasons we do not believe that the repeatability evaluation should play a role in acceptance decisions.

I have collected some rough statistics about past HSCC conferences. In HSCC 2013:

- 30 full papers.
- 20 had a significant computational component, 1 had a trivial compuational components, and 9 had no computational component.
- Among those with a signficant computational component: 7 were Matlab / Simulink, 13 were another specified language / environment (of which 2 also used Matlab), 2 had significant hardware components, and 6 made no statement about the computational environment.
- Only 3 provided links to code / data (they were not tested).

In HSCC 2012:

- 28 full papers.
- 20 had a significant computational component, 4 had trivial compuational components, and 4 had no computational component.
- Among those with a signficant computational component: 10 were Matlab / Simulink, 6 were another specified language / environment (of which 3 also used Matlab), 1 had a significant hardware component, 1 involved significant data collection and 6 made no statement about the computational environment.
- Only 4 provided links to code / data (they were not tested).
- 1 of the 2 tool papers provided links to code.

In HSCC 2011:

- 31 full papers.
- 21 had a significant computational component, 2 had trivial compuational components, and 8 had no computational component.
- Among those with a signficant computational component: 6 were Matlab / Simulink, 5 were another specified language / environment, 1 had a significant hardware component, 1 involved significant data collection and 8 made no statement about the computational environment.
- Only 3 provided links to code / data (they were not tested).
- All 4 tool papers provided links to code / data.

The critical statistics are that a strong majority of papers involve a computational component, but only a tiny fraction even attempt to allow detailed scrutiny of that component. Based on anecdotal evidence, I believe the lack of openness is not a purposeful or even

desirable outcome, but is rather driven by a lack of incentive and models so that students can learn the benefits of well managed code and data. This proposal seeks to address those deficiencies.

Some motivation for students / postdocs to take part in the REC: they get their name in the proceedings as REC members, they learn more about managing code and data by seeing how other people do it, they have the opportunity to see HSCC papers and the RPs a few months early, and the REC can serve as a training ground for the PC.

In order to ensure consistency in the first few years, I am willing to commit to being REC chair for two years and then finding and training a replacement in the third year if the process proves successful.

This document is a draft proposal, and feedback is welcome.

References

[1] N. Barnes and D. Jones. Clear climate code: Rewriting legacy science software for clarity. *IEEE Software*, 28(6):36–42, 2011. doi:10.1109/MS.2011.113.

[2] Steve Blackburn and Matthias Hauswirth. URL: http://evaluate.inf.usi.ch/artifacts/aea.

[3] Philippe Bonnet, Stefan Manegold, Matias Bjørling, Wei Cao, Javier Gonzalez, Joel Granados, Nancy Hall, Stratos Idreos, Milena Ivanova, Ryan Johnson, David Koop, Tim Kraska, René Müller, Dan Olteanu, Paolo Papotti, Christine Reilly, Dimitris Tsirogiannis, Cong Yu, Juliana Freire, and Dennis Shasha. Repeatability and workability evaluation of SIGMOD 2011. *SIGMOD Record*, 40(2):45–48, June 2011. doi:10.1145/2034863.2034873.

[4] D.L. Donoho, A. Maleki, I.U. Rahman, M. Shahram, and V. Stodden. Reproducible research in computational harmonic analysis. *Computing in Science Engineering*, 11(1):8–18, 2009. doi:10.1109/MCSE.2009.15.

[5] Jo Erskine Hannay, Carolyn MacLeod, Janice Singer, Hans Petter Langtangen, Dietmar Pfahl, and Greg Wilson. How do scientists develop and use scientific software? In *ICSE Workshop on Software Engineering for Computational Science and Engineering*, pages 1–8, 2009. doi:10.1109/SECSE.2009.5069155.

[6] Michael A. Heroux and James M. Willenbring. Barely sufficient software engineering: 10 practices to improve your CSE software. In *ICSE Workshop on Software Engineering for Computational Science and Engineering*, pages 15–21, 2009. doi:10.1109/SECSE.2009.5069157.

[7] Darrel Ince. The Duke University scandal – what can be done? *Significance*, 8(3):113–115, August 2011. doi:10.1111/j.1740-9713.2011.00505.x.

[8] Zeeya Merali. ...error...why scientific programming does not compute. *Nature*, 467:775–777, 2010. doi:10.1038/467775a.

[9] Greg Miller. A scientist's nightmare: Software problem leads to five retractions. *Science*, 314:1856–1857, 22 December 2006. doi:10.1126/science.314.5807.1856.

[10] Heather A. Pirowar, Roger S. Day, and Douglas B. Fridsma. Sharing detailed research data is associated with increased citation rate. *PLoS ONE*, 2:e308, 2007. doi:10.1371/journal.pone.0000308.

[11] Prakash Prabhu, Thomas B. Jablin, Arun Raman, Yun Zhang, Jialu Huang, Hanjun Kim, Nick P. Johnson, Feng Liu, Soumyadeep Ghosh, Stephen Beard, Taewook Oh, Matthew Zoufaly, David Walker, and David I. August. A survey of the practice of computational science. In *State of the Practice Reports, Supercomputing*, pages 19:1–19:12, 2011. doi:10.1145/2063348.2063374.

[12] P. Vandewalle, J. Kovacevic, and M. Vetterli. Reproducible research in signal processing. *IEEE Signal Processing Magazine*, 26(3):37–47, May 2009. doi:10.1109/MSP.2009.932122.

[13] Greg Wilson, D. A. Aruliah, C. Titus Brown, Neil P. Chue Hong, Matt Davis, Richard T. Guy, Steven H. D. Haddock, Katy Huff, Ian M. Mitchell, Mark Plumbley, Ben Waugh, Ethan P. White, and Paul Wilson. Best practices for scientific computing. Posted October 2012 to Arxiv for feedback, submitted June 2013 to PLOS Biology, accepted October 2013. URL: http://arxiv.org/abs/1210.0530.