Review: "CoRR — The Cloud of Reproducible Records"

Fernando Chirigati[1]

[1]Computing in Science and Engineering

April 28, 2020

This is a review of manuscript CiSESI-2018-02-0016 submitted to Computing in Science & Engineering: "CoRR — The Cloud of Reproducible Records" (Congo, Traoré, Hill and Wheeler, 2018).

Overview

The paper presents CoRR (Cloud of Reproducible Records), a Web platform for storing and managing records from different tools that create snapshots of computational environments for reproducibility purposes. The authors refer to these tools as CVC (Computation Version Control) tools. In a nutshell, these tools capture the state of the environment in which computational environments are run (e.g.: OS and hardware information, library dependencies, system variables, etc.), in addition to the code and the data. Examples of CVC tools include Sumatra, ReproZip, and CDE. The authors argue that CVC tools are facing major issues in adoption, and that one of the main reasons is related to the lack of a Web interface for sharing and managing CVC records (similar to what GitHub or BitBucket do for SVC tools). CoRR was designed to fill this gap and to facilitate the integration among these tools by providing a common management platform.

A common platform for these tools is indeed interesting and useful for reproducibility. However, the contributions of the manuscript are still not clear. More details are provided in the next section, but here is a summary of the main issues:

1. The differences between CoRR and existing data repositories need to be made more clear.
2. The name CVC is misleading.
3. The way that the metadata is stored in the platform is not clear.
4. Diffs in the platform are manual rather than automatic, and there is no discussion on the challenges related to these diffs.
5. Related work about provenance, workflows, and repositories are missing.

My recommendation is "*Author Should Prepare A Major Revision For A Second Review*".

Detailed Review

I should note that I tried to get access to the platform, but I wasn't able to (no confirmation email was sent as of yet). I also tried to use the search feature in the main Website, but it keeps loading after pressing the return key and no results are returned.

1. The differences between CoRR and existing data repositories need to be made more clear.

CoRR is a repository of computation records. But what makes it different from other data repositories? This is still not clear to me.

It seems that one of the main benefits of CoRR is the ability of exposing the metadata that the CVC tools capture, and allowing these to be queryable. For instance, in a regular data repository, if I want to search for projects that used scikit-learn, I would only be able to find such information if it were present in the description of the artifacts. On the other hand, in CoRR, one could make this information automatically available for querying, since tools like Sumatra or ReproZip capture such dependencies.

Are these metadata indeed queryable in CoRR? If yes, this is a major benefit and should be made more explicit in the paper. In general, the paper would benefit from a section where authors explicitly discuss the main differences between CoRR and existing data repositories when it comes to CVC tools, i.e., why would someone choose to use CoRR and not any of the existing repositories?

2. The name CVC is misleading.

CVC stands for Computation Version Control, but neither ReproZip nor CDE do version control: they do create a snapshot of the computation, but they do not have a mechanism for version control. The authors seem to be referring to tools that capture provenance related to the computational environment, but not necessarily that provide version control, so the nomenclature should be changed.

3. The way that the metadata is stored in the platform is not clear.

The section "Adaptive and Open Database Model" was not clear enough to me. How are different metadata (from different tools) stored in a single data store? The authors do present the MongoDB's models, but there are no details on how different metadata are integrated into a single model. And why not use some representation like PROV (https://www.w3.org/TR/prov-primer/) for integrating the models?

4. Diffs in the platform are manual rather than automatic, and there is no discussion on the challenges related to these diffs.

At the end of the paper, the authors discuss the concept of diff as a way to tell whether a computation X is a replicate, a repeat, or a reproduction of a computation Y. This is a really cool feature, but it looks like users in CoRR need to define such diffs manually, which is certainly not scalable when dealing with hundreds

of computations. Automatically figuring out if two computations are similar in terms of reproducibility is challenging, in particular if they were captured by different tools. But this is certainly a very useful feature for a repository such as CoRR. I was expecting at least a more detailed discussion about this.

5. Related work about provenance, workflows, and repositories are missing.

Capturing provenance from computations is certainly not a novel topic, and some references are missing. Authors should acknowledge scientific workflow management systems (e.g.: Taverna (Missier et al., 2010), Kepler (Ludäscher et al., 2006), and VisTrails (Freire et al., 2011)), since they are known for capturing provenance from experiments (Davidson and Freire, 2008) (Freire et al., 2008). In terms of representing provenance information, the authors should take a look at PROV (`https://www.w3.org/TR/prov-primer/`).

There are also other tools that capture provenance from computational environments: although some might not be widely adopted, it is worth mentioning them. These are: PTU (Pham et al., 2013), CARE (Janin et al., 2014), Arnold (Devecsery et al., 2014), and noWorkflow (Murta et al., 2015).

I also recommend taking a look at the related work section of these papers to see if there are additional relevant references as well.

Finally, since CoRR is a repository, it is important to acknowledge existing collaborative data repositories, e.g.: Dataverse, figshare, OSF, etc. Again, as I mentioned before, it is important to provide a detailed comparison against these repositories.

Additional Comments

- The authors mention in the Introduction that the lack of a platform for storing and managing computation records is probably one of the main reasons for the slow adoption of CVC tools. However, there is no evidence for that. Is there any reference or further study that the authors can provide to back this up? The motivation is not clear.
- There are other tools (including noWorkflow) supported by CoRR (Figure 3). Why aren't these tools mentioned in the paper?
- Using forked repositories from the existing tools might not be ideal. It might be hard to keep updating these repositories as the original ones keep changing. For instance, a user might need the latest features of Sumatra, but the CoRR-related Sumatra repository might be a few commits behind. Are there any thoughts on that? Why not just have a standalone CoRR software that reads project information from Sumatra / ReproZip / CDE and uploads data to the platform?

Disclaimer

For the sake of transparency, it should be noted that I am one of the developers for ReproZip, one of the software tools reviewed in the manuscript and used in the case study.

References

Susan B. Davidson and Juliana Freire. Provenance and scientific workflows. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. ACM Press, 2008. doi: 10.1145/1376616.1376772. URL https://doi.org/10.1145%2F1376616.1376772.

David Devecsery, MIchael Chow, Xianzheng Dou, Jason Flinn, and Peter M. Chen. Eidetic Systems. In *11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14)*, pages 525–540, Broomfield, CO, 2014. USENIX Association. ISBN 978-1-931971-16-4. URL https://www.usenix.org/conference/osdi14/technical-sessions/presentation/devecsery.

Juliana Freire, David Koop, Emanuele Santos, and Cl Silva. Provenance for Computational Tasks: A Survey. *Computing in Science & Engineering*, 10(3):11–21, may 2008. doi: 10.1109/mcse.2008.79. URL https://doi.org/10.1109%2Fmcse.2008.79.

Juliana Freire, David Koop, Emanuele Santos, Carlos Scheidegger, Claudio Silva, and Huy Vo. Chapter: VisTrails. *The Architecture of Open Source Applications*, 2011.

Yves Janin, Cédric Vincent, and Rémi Duraffort. CARE, the comprehensive archiver for reproducible execution. In *Proceedings of the 1st ACM SIGPLAN Workshop on Reproducible Research Methodologies and New Publication Models in Computer Engineering - TRUST '14*. ACM Press, 2014. doi: 10.1145/2618137.2618138. URL https://doi.org/10.1145%2F2618137.2618138.

Bertram Ludäscher, Ilkay Altintas, Chad Berkley, Dan Higgins, Efrat Jaeger, Matthew Jones, Edward A. Lee, Jing Tao, and Yang Zhao. Scientific Workflow Management and the Kepler System: Research Articles. *Concurr. Comput. : Pract. Exper.*, 18(10):1039–1065, aug 2006. ISSN 1532-0626. doi: 10.1002/cpe.v18:10. URL http://dx.doi.org/10.1002/cpe.v18:10.

Paolo Missier, Stian Soiland-Reyes, Stuart Owen, Wei Tan, Alexandra Nenadic, Ian Dunlop, Alan Williams, Tom Oinn, and Carole Goble. Taverna Reloaded. In *Lecture Notes in Computer Science*, pages 471–481. Springer Berlin Heidelberg, 2010. doi: 10.1007/978-3-642-13818-8_33. URL https://doi.org/10.1007%2F978-3-642-13818-8_33.

Leonardo Murta, Vanessa Braganholo, Fernando Chirigati, David Koop, and Juliana Freire. noWorkflow: Capturing and Analyzing Provenance of Scripts. In *Lecture Notes in Computer Science*, pages 71–83. Springer International Publishing, 2015. doi: 10.1007/978-3-319-16462-5_6. URL https://doi.org/10.1007%2F978-3-319-16462-5_6.

Quan Pham, Tanu Malik, and Ian Foster. Using Provenance for Repeatability. In *Presented as part of the 5th USENIX Workshop on the Theory and Practice of Provenance*, Lombard, IL, 2013. USENIX. URL https://www.usenix.org/conference/tapp13/using-provenance-repeatability.