Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2017-242-AC1, 2018

Geoscientific Model Development Discussions

Interactive comment on "Best practice regarding the three P's: profiling, portability and provenance when running HPC geoscientific applications" *by* Wendy Sharples et al.

Wendy Sharples et al.

w.sharples@fz-juelich.de

Received and published: 10 January 2018

We thank the anonymous reviewer for taking the time to read this paper thoroughly and providing the authors with constructive and thoughtful feedback. Addressing this feedback has greatly improved the paper. I have provided a pdf file of the text below also with our responses in green which might be easier to read.

Addressing Reviewer #1's concerns:

Title: Best practice regarding the three P's: profiling, portability and provenance when running HPC geoscientific applications Author(s): Wendy Sharples et al. This paper

presents a utility for controlling the execution and initial evaluation of an application (the ParFlow model) running in a (primarily) HPC framework. There are two levels to the framework described: the "run control framework" (or RCF) which itself utilises a more generic JUBE benchmark framework as a workflow engine. Essentially these provide a method of systematically defining, running and analysing some benchmarks - the authors also suggest it would be suitable for use for production simulations as well. The framework isolates multiple runs using parameters configured at set up time and keeps all the data produced along with a set of reports, allowing parameter sweeps with automated isolation of the various results (termed "provenance tracking" in the paper). A case study is presented utilising the framework to examine weak scaling (increasing the domain size) for the ParFlow model.

This paper is a difficult read, because there are three different strands within it: motivation, tooling, and the results of using the tools. They are well isolated by the sections, but each is somewhat unsatisfying on its own, and the links between are not as strong as I would like to see in a GMD journal paper. As it stands I do not think it is fit for full publication in GMD, but I think it could be made so with some reworking to make the material more accessible and relevant to the GMD audience.

There is some good material in the motivation, but it falls uncomfortably between being either a complete description of the portability, performance and reproducibility issues associated with geo-scientific modelling or an introduction to those elements for which the tools discussed later are either well suited or applied. It would be stronger if it were the latter.

>Our aim was to introduce those elements for which our RCF is a solution. We have therefore provided a better and more complete explanation of exactly what the RCF is, what its capabilities and functions are and how it is a best practice approach to aid portability, profiling and provenance tracking.

The discussion of the tooling itself is incomplete in important details, yet full of detail

(like the XML files in the appendix) which cannot be easily consumed by the reader because of a lack of appropriate explanation. There is no discussion of why this tool is any different from any other tool (e.g. what are the strengths and weaknesses with respect to the two workflow tools introduced in section 1)?

>This was an oversight on our part. We have now included a discussion of the relevant strengths and weaknesses of JUBE, ecFlow and cylc and have stated the reasons for choosing JUBE and that the RCF could be adapted for other workflow engines. We have also tried to better describe what the RCF is separate from the workflow engine used (JUBE) and how they are integrated with each other.

The results of the analysis of ParFlow are probably the strongest and most interesting parts of the paper, but because of the layout of the paper, introductory material such as the definitions of load balance, are mixed in with results and interpretation. I would rather this section had been organised to correspond to the (very useful) list of "health checks" which begins on the bottom of page 6. It might have been that the relevant definitions (equations 1 to 4) could have appeared in section 2, since that's where these issues are first introduced. The results and scientific consequences could then be clearly identified in 3.4.

>Thanks for this great suggestion. We have moved these definitions up to section 2, and mapped each health check step back to the results presented and have provided a more in depth description of the profiling tools used and the function of the RCF in automating the health check workflow.

Addressing the specific comments:

1. I do not believe the title fairly reflects the material of the paper. The paper is not about best practice, it is primarily about one workflow/benchmarking application, although it does list elements of best practice and motivate some of the issues.

We wanted to present the RCF as a best practice approach, but we accept that the

title does not totally reflect the material presented. We have updated the title to: A run control framework to streamline profiling, porting and tuning, simulation runs and provenance tracking of geoscientific applications

2. The paper begins with motivation with a selective list of references for how increased HPC might be used. The list is somewhat different from that usually presented which normally now includes increased use of data assimilation alongside increasing complexity, domain (spatial or temporal), resolution and ensemble size. It would be good to see this list inclusive of data assimilation and temporal extent and without quite so many references which don't add much value (there are so many it appears to be an attempt to be exhaustive, but it is clearly not exhaustive, better to have few or no exemplars than three or four each, because one is left wondering "why *these* ones"?).

This is a good point. We have updated the reference lists so that there are no more than 3 and e.g. is used more often to indicate the references are a subset of a broader scope of research (1st paragraph)

3. There is then some material on the upcoming difficulties with performance portability which adds to the motivation, but the implication is that these are issues which can be solved by optimisation. In particular the paragraph beginning on line4 of page 3 begins by recognising the massive investments required to get performance, then implies that this investment can leverage analysis of existing codes using benchmarking tools such as the RCF/JUBE one discussed here. I think this section would be stronger if there was a clean separation between the aspirations of parameter sweeping and performance analysis, which is primarily about optimisation, and that of massive structural reorganisation of code such as was involved in Leutwyler et al. This is not to denigrate the importance of the former, but just to realistically recognise the scope. As written, the paper overstates it.

Yes this is true. We meant to say that the RCF can streamline the investment as bottlenecks can be quickly identified because it has the relevant profiling tools integrated

within it to automate the performance engineering approach. We have tried to clarify/add to this explanation in the paragraph mentioned in page 3 and we refer back to this automation in section 3.

4. In the context of scope it would also be useful to identify where the tool might have significant limitations, e.g, where it could interfere with other configuration and workflow managers (or be interfered with). This is not to suggest that the tool is not useful, or even powerful, for a particular class of problems; just that like all solutions, it almost certainly has limited applicability. It would be a service for potential users for the authors to provide some clarity on any known scope issues.

Good suggestion. In the section on JUBE 2.1 paragraph 2 discussion of limitations have been added to and also what is common between JUBE, cylc and ecFlow is stated. In addition the advantages/disadvantages of each workflow engine is discussed along with an explanation of why we have chosen JUBE.

5. I think the paper confuses key issues around reproducibility. The implication of the discussions about reproducibility on page 3 and section 3.3 is that "if only the relevant parameters were documented, simulations would be reproducible". While this is undoubtedly *necessary*, it is far from *sufficient*, Baker et al. doi.org/10.5194/gmd-8-2829-2015 discuss the issue of ensuring that the science remains the same when hardware and software environments change. This paper would be stronger for identifying the distinction between these different issues of reproducibility and linking them to Irving's discussion and prior literature.

Our RCF actually goes much further than just documenting the relevant parameters. The whole simulation can be rerun as the code, the forcing data, the environment: source file with modules to be loaded, including a list of dependencies their versions, all extraneous scripts including job submission scripts and a complete model description are bundled together. If a user were to untar an output directory, they would be able to compile and rerun the experiment with JUBE using the RCF XML file in the directory,

on the same machine and obtain the same results. We have improved the description on page 3 and section 3.3. Also we have mentioned that there are some things beyond the users control such as different hardware and compilers when porting models and ways to ensure the science remains the same.

6. This might address the issue that there is only a cursory discussion of the issues associated with porting ParView to JUQUEEN - indeed, one might have expected the use of this framework to help with that process. "It was found that the optimal flags which did not compromise accuracy" with accuracy determined "to six figures". This reviewer has no idea what they meant by "accuracy" in this context, and the cursory argument suggests that important issues around solution stability were not investigated (despite the motivation being reproducibility).

Yes the test was not very well explained. It was briefly explained in section 2.3. We have expanded this section and made reference to this section in section 3.2.

7. In the discussion of the tools itself, the overall workflow is well described (Figure 1 etc and the excellent list provided for the "health examination"), but the discussion of the tool provides names of files, and then exemplar files (in XML, in the appendices). It's not clear at all how and why this framework is better than a bunch of scripts with input files - it would be considerably stronger if there was some discussion of how the tool exploits the XML files (is there a semantic structure inherent in the files beyond that provided by the use of XML to control syntax)? It might be that this is what the description JUBE reference provides, but I was unable to access the description of JUBE hidden behind a paywall. Some kind of discussion about how the XML content links to JUBE actions would be helpful. In any case, I recommend removal of the XML files in the appendix, on their own they are inscrutable and provide little value. However, if they were provided in a repo with documentation as to function, they would provide useful complementary material.

Good suggestion. We have moved the description of the RCF from the appendix to

section 2 to help provide a fuller understanding of exactly what it is. We have tried to emphasize the links between the XML content and JUBE actions in section 2.2 to explain how the RCF builds the overall configuration XML file to be run by JUBE. The xml files are provided in a tarred up directory submitted with this manuscript. There are readme files plus documentation in each script as to what it's functionality is. To that end we have removed the xml files from the appendix.

8. The bulk of the case study shows the ability of the tools to generate information to understand the performance of the ParView model on this platform, and introduces some of the plans to alleviate performance bottlenecks (such as Adaptive Mesh Refinement). However, I did not fully understand the argument as to why this is the obvious next step from the current arrangement where all cells know about all other cells (why)?. This piece of the argument was another place where I felt that there was blurring together in this paper around issues of performance portability (optimising for a target architecture, but not changing the science), versus algorithmic improvements in performance (which involve changing the science).

We agree that the integration with p4est was not very clear. This is a case of an algorithmic improvement which does not change the science. ParFlow coupled to p4est means that p4est is now the parallel mesh manager, currently focusing on uniform meshes. The approach was minimally invasive and preserves most of ParFlow's data structures, the configuration system and the setup and solver pipeline. We have now explained this in section 3.4 and also mentioned the state of the current mesh manager in section 3.3.

9. Somewhere in the paper there needs to be some comparison to prior art and other similar tools which may address some part of the scope of these tools. (The description of GMD Development and Technical papers states: "Development and technical papers usually include a significant amount of evaluation against standard benchmarks, observations, and/or other model output as appropriate.") While, the key word may be *usually*, in the context of *this* paper I think there should be a section similar to the

"RelatedWork" section that appears in many computer science and software engineering papers covering relevant strengths and weaknesses.

There is related work in terms of scaling and profiling ParFlow, which has been reported in references Kollet et al. 2010 and Gasper et al. 2014 for example. There is a report on comparisons between workflow engines cylc and ecFlow in Manubens-Gil 2016 which is now mentioned in our manuscript. Our reasons for choosing JUBE over cylc and ecFlow are now outlined clearly in the manuscript, where the main reason for choosing JUBE is that it is the workflow engine most compatible for use on JSC machines. However the RCF itself is a new development to streamline scientific software profiling, porting and tuning, running production runs and to aid reproducibility. To date, there are no “standard” benchmarks as such.

10. Finally, I note that the code is apparently available on request, but I think it will not get significant uptake without being both open source and having an open development process. Neither are required for GMD publication, I only mention these factors to encourage the authors to build a community around what looks like a useful tool for a certain class of problems.

A tarball has been included with the manuscript which the users can use with JUBE and any version of ParFlow with the test case described in the paper straight away. This should be enough to get a new user familiarized with the RCF enough to add to it for their own purpose. The reason why the whole RCF code is not open source is because the real world models included are developed by different scientists, many of which have not yet been published.

Please also note the supplement to this comment:
https://www.geosci-model-dev-discuss.net/gmd-2017-242/gmd-2017-242-AC1-supplement.pdf

---

Interactive comment on Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-242,

2017.