PDSW'20 (co-located with SC20) Artifact Submission Guidelines

As part of the submission process of PDSW'20, authors can optionally submit an artifact description (AD) associated with their article submission. The AD includes a field for one or more links to data (zenodo, figshare, etc.) and code (github, gitlab, bitbucket, etc.) repositories. This document provides guidelines on how to structure the artifact that will be placed in the code repository. While it is not required to use this exact same internal organization for the artifact, we encourage authors to follow these guidelines as it will make it easier to the reviewing committee and readers of the paper in the future.

Outline of the Digital Artifact

The subdirectories in the project folder can contain the necessary scripts, data, figures and tables created during the study, along with a description of what is contained in the artifact and instructions of how to run the code provided.

The digital artifact can contain the following:

```
.
└── my_repo_folder
    ├── compile
    ├── doc
    ├── figures
    │   ├── output
    │   └── scripts
    ├── input
    └── run
        ├── output
        └── scripts
```

The `doc` directory

The `doc` directory can contain a README. This can be in Markdown or plain text. This will serve as a guide to understand what the contents of the artifact are and how to use them. It can include:

- A description of the contents of the artifact
- Instructions on how to run/view/use each part of the artifact

- How to compare the artifact's output with the output presented in the original paper(s).

The doc directory can also contain the final manuscript (pre-print) in .pdf format

The `compile` directory

- The `compile` directory can contain a "compilation" script or set of scripts. These scripts can produce the binaries and any other libraries used by the next "run" script(s). The scripts can build all major dependencies that are required by the application under test. The output of this script can be the binaries used in the original paper (s).
- Include a README in the `compile` directory that gives a brief explanation of how the script(s) work, including:
    - compiler information
    - Which software dependencies are being built by the script vs. supplied by the OS or .rpms
    - Any comments on what changes or modifications were done in order to compile on the new environment.
- Include in the main README in the `doc` directory how to compile the application (i.e. how to invoke the script(s))

The `run` directory

The `run` directory can contain two subdirectories, a `scripts` subdirectory and an `output` subdirectory.

`scripts` subdirectory

The `scripts` subdirectory can contain a "run" script. This script actually runs the application for the given inputs and creates the logs/output and results used in the "figures" script. The README can describe how the "run" script is used including:

- Options and environment variables, etc.
- Environment at run time (e.g. `env` output)
- How runs resulting from the script might differ from a re-execution of experiments contained in the original paper(s), for example input data sets, number of processes, etc.
- Indicate how timings were measured.

`output` subdirectory

The `output` subdirectory can contain logs/output resulting from the “run” script. These result files are those used by the “figures” script. You could, for example, use the `output` subdirectory as the target for the output of the scripts in the `run/scripts` subdirectory.

The `figures` directory

The `figures` directory can contain two subdirectories, a `scripts` subdirectory and an `output` subdirectory.

`scripts` subdirectory

The script subdirectory can contain a “figures” script. This script takes the logs and output from running the “run” script in order to create the new environment’s version of any figure being reproduced, or new ones being generated. The README can detail how the “figures” script is run, and what tools and software were utilized.

`output` subdirectory

The figures/output directory can contain tables, charts and figures that result from running the “figures” script. The README can include:

- a brief description of the results seen in the output plots (this commentary can be largely taken from the report text)
- comment the speedups/performance and how others can compare these when they re-execute the experiment on their machine/platform.

Tools to prepare a submission

You may write the scripts in a shell scripting language or Python, or any other scripting language you feel comfortable writing scripts in. Automation tools can also optionally be used to help prepare the artifact submission. If one of these are used, please include any relevant configuration files or instructions with the report and artifact.

- Spack: a flexible package manager that supports multiple versions, configurations, platforms, and compilers
- EasyBuild: software build and installation framework to manage scientific software on HPC systems in an efficient way
- CK: Guide to help you create, run and pack your SCC workflow
- Popper: Guide on creating a workflow from an existing set of scripts