**Pegasus and Makeflow on Stampede**

**Instructions to install Makeflow via tar file:**

$ cd $HOME

$ wget http://ccl.cse.nd.edu/software/files/cctools-5.1.1-source.tar.gz

$ tar xvzf cctools-5.1.1-source.tar.gz

$ cd cctools-5.1.1-source

$ ./configure

$ make

$ make install

$ cd $HOME

$ echo "PATH=/$PATH:\$HOME/cctools-5.1.1/bin" >> ~/.bashrc

$ source ~/.bashrc

**Initial setup of Makeflow:**

Write a makefile defining all the rules of the workflow. Rules consist of a target followed by a colon and any dependencies, with commands on the next line after an indentation.

*Example:*

output.txt: mysim.exe calib.data

./mysim.exe -c calib.data -o output.txt

Unlike Make, Makeflow scripts cannot contain “empty” rules such as *clean*. All rules must have dependencies strictly defined.

Note: Makeflow Makefiles use different syntax than Make Makefiles.

**To run a job through SLURM with Makeflow:**

*Example:*

$ makeflow -T slurm -B "-p normal -t 1" example.makeflow

Parameter –B submits options to the batch system. SLURM requires parameter –p to specify which queue to submit through, and parameter –t to specify time allocated for the job to run. More arguments are available to be added between the quotes. More on arguments can be found in the manual (link included below).

**To run a job through Work Queue with Makeflow:**

1. Initialize work queue.

*Example:*

$ makeflow -T wq -p 0 -N PROJECT-$USER --work-queue-preferred-connection by\_hostname example.makeflow

1. Start workers:
   1. For running locally:

$ work\_queue\_worker -N PROJECT-$USER

* 1. For batch using SLURM:

$ slurm\_submit\_workers -N PROJECT-$USER -p "-t 1:00 -p normal" 2

Again, reference Makeflow manual for more arguments.

**Some notes on Makeflow vs. Pegasus:**

Makeflow has an advantage over Pegasus in that for a user experienced with workflows it can be initially easier to learn. It is built on 40-year-old traditional Make, so many users already familiar with Make should find the transition to Makeflow fairly smooth. Makeflow still uses Makefiles, with the main difference being that it does not rely on a distributed file system. Instead, it uses the dependency information included in the Makefile to send data to remote jobs. Because of this, users must be careful in writing Makefiles to set up every dependency precisely, but once correctly written this ensures correct job execution and resource management.

Makeflow documentation has good simple examples, but since it is built on Make, there is not extensive instruction on learning Makeflow. If a user is not already familiar with Make, they would need to start by learning Make and then learn the modifications for Makeflow. There is extensive documentation on Make (link available below). While Pegasus has extensive documentation and training examples, Makeflow training is adequate only if the user is familiar with Make.

While Makeflow initial setup and learning is easier than Pegasus, it does not have as many helpful features as Pegasus. For example, it uses the Makefile log to recover after a failure and to record how and when every task is run, but does not support provenance queries as Pegasus does. However, Pegasus requires a lot more setup to run, so for simpler jobs or users accustomed to Make, Makeflow is likely a better choice.

**Useful links:**

Makeflow manual:

<http://ccl.cse.nd.edu/software/manuals/man/makeflow.html>

Make manual:

<https://www.gnu.org/software/make/manual/make.html>

A brief summary of the development of Makeflow:

<http://dthain.blogspot.com/2009/07/make-as-abstraction-for-distributed.html>

An introduction to Makeflow by the developer:

<http://dthain.blogspot.com/2012/02/why-makeflow-works-for-new-users.html>

A Makeflow tutorial with example data:

<https://sites.google.com/site/xsedeworkflows/makeflow-tutorial#TOC-Running-Makeflow-in-Stampede>

**Instructions to install Pegasus via tar file:**

$ wget http://download.pegasus.isi.edu/pegasus/4.6.1/pegasus-binary-4.6.1-x86\_64\_rhel\_7.tar.gz

$ tar zxf pegasus-binary-4.6.1-x86\_64\_rhel\_7.tar.gz

$ echo "PATH=/$PATH:\$HOME/pegasus-4.6.1/bin" >> ~/.bashrc

$ source ~/.bashrc

NOTE: It may be necessary to add Pegasus to your Python path:

$ echo "PYTHONPATH=\$PYTHONPATH:\$HOME/pegasus-4.6.1/lib64/python2.7/site-packages" >> ~/.bashrc

$ source ~/.bashrc

**Initial setup of Pegasus:**

1. **Write a DAX generator file** defining the directed graph. This requires initialization of an abstract ADAG object, all filenames (input, output, and intermediate files), and each job as a Job object. This file also requires any arguments, an indication of which files are used for input and output, and indication of all dependencies. Finally, the DAX generator writes an XML file to sys.stdout.
2. **Create catalogs**. This includes:
   * The **site catalog**: an XML file named sites.xml describing the sites where the jobs will be executed
   * The **transformation catalog**: a text file named tc.txt describing all of the executables used by the workflow, where they are located, the architecture and OS they are compiled for, and any other needed information
   * The **replica catalog**: where to locate each of the input files needed for the workflow
3. **Write configuration file pegasus.properties** which contains properties such as where to find the catalogs, what to use to transfer data, and the name of the application.

**To run a job:**

1. Generate workflow with the DAX generator.

*Example:*

$ ./dax-generator.py > workflow.dax

1. Run pegasus-plan to compile everything together.

*Example:*

$ pegasus-plan \

--conf pegasus.conf \

--sites sdsc-gordon,tacc-stampede \

--output-site local \

--dir work \

--dax workflow.dax

1. Run the workflow:

$ pegasus-run /home/(path)

*“Path” is the path to the submit directory where all of the files required to submit and monitor the workflow are stored*

**Notes on Pegasus vs. Makeflow:**

Without explicit direction, Pegasus is difficult to install on Stampede because there are many sets of installation instructions but it is hard to find which instructions are appropriate for Stampede. Given the steps above, it is easy to install. There are also some Atmosphere images set up with Pegasus (links included below) to avoid installing directly on Stampede.

Pegasus has a steeper learning curve than Makeflow for two reasons: it is new software rather than being built on an old system like Make, and it has many more features, requiring more setup and user education.

While Makeflow only requires one Makefile script, Pegasus requires several components as described in the setup outline. A link is included below containing simple examples of the files required for a Pegasus workflow. However, Pegasus provides good documentation on how to set up these files, and once a user becomes familiar with the process it may be worth the extra work for the many features Pegasus offers. Links are included below to the Pegasus manual and tutorials.

One example of a Pegasus feature that Makeflow does not have is the way jobs are launched through a kickstart process. This process captures runtime provenance data, which is stored in a database and can return summaries via Pegasus tools or through SQL queries. Another example is the easy-to-use visual analysis (command pegasus-service) offered by Pegasus to users with a GUI.

**Useful links:**

Atmosphere images set up for Pegasus:

<https://atmo.iplantcollaborative.org/application/images/699>

<https://atmo.iplantcollaborative.org/application/images/720>

<https://atmo.iplantcollaborative.org/application/images/737>

Pegasus User Guide:

<https://pegasus.isi.edu/documentation/index.php>

Simple workflow examples in Pegasus:

<https://pegasus.isi.edu/documentation/examples/>

Example of using Pegasus on XSEDE including Stampede:

<https://sites.google.com/site/xsedeworkflows/pegasus-tutorial>

Tutorial using a VM with Pegasus and tutorial data preloaded:

<https://pegasus.isi.edu/documentation/tutorial.php>

**Direct comparison of qualities in Makeflow and Pegasus:**

|  |  |  |
| --- | --- | --- |
|  | **Makeflow** | **Pegasus** |
| **Portability** | Does not reference any particular batch system or distributed computing technology, so existing workflows can be easily moved between different systems. | Neutral DAX scripts can be run on a single system or across a set of different resources. |
| **Performance** | Makeflow treats file dependencies very literally. This improves workflow performance by setting up the right environment, verifying correct execution, and managing resources. | The Pegasus mapper reorders, groups and prioritizes tasks to increase workflow performance. |
| **Scalability** | Makeflow can be run very easily on a single machine, or can use SLURM directly or with Work Queue to run on Stampede. | Pegasus can scale the size of the workflow and the resources the workflow is distributed over. For very large jobs, it allows “subworkflows” to allow compartmentalization of jobs. |
| **Reliability** | Can pass parameter –r when starting a job so Makeflow will automatically retry failed tasks. Outputs a logfile of debugging statements. | Jobs and data transfers are automatically retried when failed. Has a tool Pegasus-analyzer to debug non-recoverable failures. |
| **Error recovery** | Uses the Makeflow log file to recover after a failure and for use in logging and debugging. | Tries to recover by retrying tasks and the workflow; providing checkpoints; re-mapping the workflow; using alternative data sources. If everything still fails, provides a “rescue” workflow of only what still needs to be done. |
| **Provenance** | Does not support provenance queries. Produces a log file which records how and when every task is run. Used for logging and debugging. | Jobs are launched through a kickstart process that captures runtime provenance to help in debugging. Provenance data is stored in a database and can be summarized with Pegasus tools or through SQL queries. |
|  |  |  |
| **Installation** | Easy to find instructions. Simple installation of cctools followed by about a minute to compile and install Make. | Difficult to dig through instructions to find the correct installation process for Stampede. Simple to install once instructions are acquired.  Atmosphere has some images available with Pegasus already installed. |
| **Learning curve** | Similar to Make with some modifications. Makeflow tutorials are easy to follow but are not very detailed. The user manual mostly gives an overview of the differences between Makeflow and Make, rather than extensive instructions on actually using Makeflow. In order to learn Makeflow, the user would either have to be already familiar with Make or would have to learn Make first. | Nice clear tutorials are provided by Pegasus, available preloaded on a VirtualBox VM. Pegasus has a lot more components to learn, but the website has a very nice user guide with a table of contents and instructions on each part of the process. |
| **Ease of use** | *Makes workflow creation easier through abstraction by not having to specify low-level details.*  Easy to use for users who are familiar with Make and makefiles. Does not have as many convenient built-in tools as Pegasus. | *Makes workflow creation easier through abstraction by not having to specify low-level details.*  Pegasus requires several components for the script – DAX generator and catalogs |