HPC Systems engineer evaluation

- *Divya Pattisapu*

Github link: https://github.com/Patty8122/pw_assignment (Code and results)

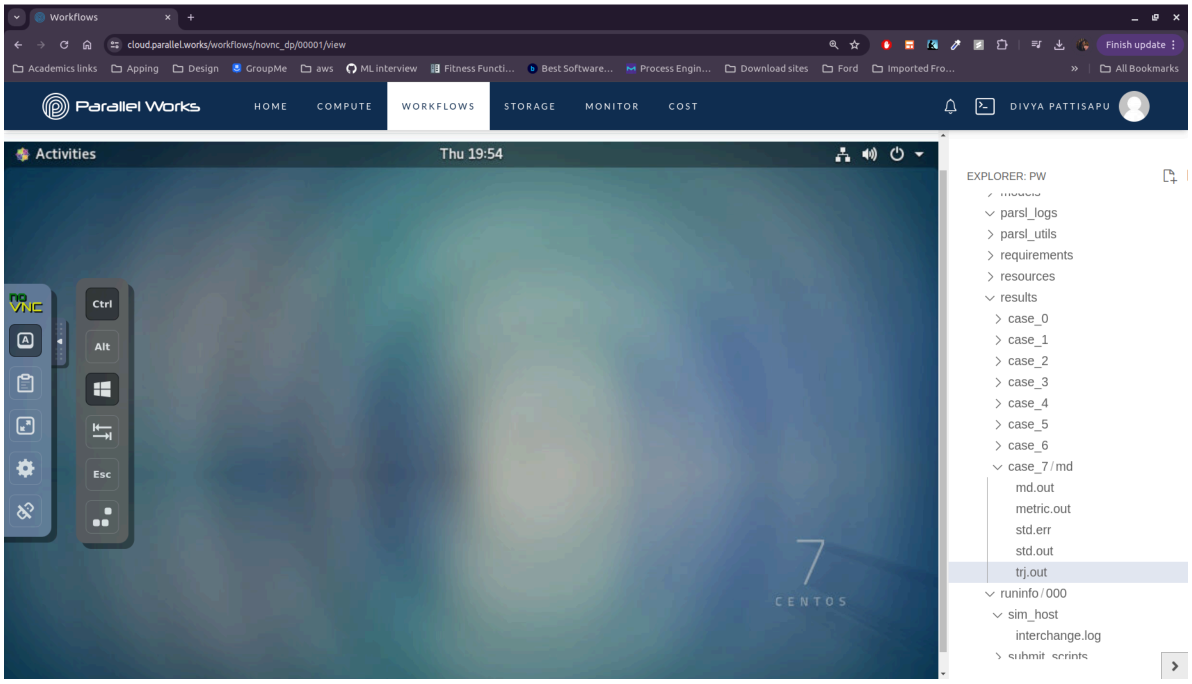
Test Part 1: Explore and evaluate the PW platform

1.1. Workflows

Summary
I have run the MDLite workflow and the resultant Job ID is 003.

Extra Credit: NoVNC
I have also launched the NoVNC workflow (Job ID is 001) in the interactive window.

Screenshot
The following is a screenshot of an interactive NoVNC workflow run.



1.2. Installation

1.2.1. OpenMPI

Summary
I installed the OpenMPI module from source into `/contrib`. For each run, I change the PATH and LD_LIBRARY_PATH of any allocated node to point to `/contrib/.openmpi/bin` and `/contrib/.openmpi/lib`

Steps taken to install and run
1. To install the package from source, run the file `/contrib/install.sh`
2. To run the hello_world program using 2 nodes, upload the attached zipped files to the home directory - `task1_2/helloworld_mpirun.sbatch` and `task1_2/mpi_hello.c`
3. Run the file `/home/dpattisapu/task1_2/helloworld_mpirun.sbatch`

Output
The generated file `/home/dpattisapu/task1_2/mpi_hello.stdout` has the output from the program. It looks like:
```
Hello world from processor dpattisapu-aws-00008-1-0001, rank 0 out of 2
processors
Hello world from processor dpattisapu-aws-00008-1-0002, rank 1 out of 2
processors
```

1.2.2. Extra credit: OpenFoam

Summary
I installed OpenFoam from source into `/contrib`. During its installation, I also had to install flex and upgrade gcc.

Steps to install and run
1. To install the package from source, run the file `/contrib/install_openfoam.sh`
2. Edit the file named `/contrib/OpenFOAM-dev/tutorialsTest/fluid/squareBend/machine.txt` by changing the index of the name of the nodes:
   dpattisapu-aws-00011-1-0001 slots=4
   dpattisapu-aws-00011-1-0002 slots=4
3. Run the commands:
   a. source /contrib/OpenFOAM-dev/etc/bashrc
   b. cd `/contrib/OpenFOAM-dev/tutorialsTest/fluid/squareBend`
   c. blockMesh
   d. decomposePar
4. Copy and paste `/contrib/OpenFOAM-dev/tutorialsTest/fluid/squareBend/squarebend_mpirun.sbatch` from the attached files if not present.

5. Run the command:
   sbatch
   `/contrib/OpenFOAM-dev/tutorialsTest/fluid/squareBend/squarebend_mpirun.sbatch`
6. The log output is present in `log-squarebend.out`

Steps to check job related information

1. scontrol show job $JOB_ID
2. sstat -j $JOB_ID

1.3 Report on experience with the platform

1.3.1 What works?

- Persisting the workflow results in the EFS storage is helpful for later access.
- The cost calculations are very helpful to keep track of expenditure.
- The workflows are easy to use.

1.3.2 What doesn't work? Were any of the features difficult to use?

- Module installation is not very straightforward. The controller node cannot set shell environment variables for other nodes. Hence, installations must be done in the EFS storage.
- The conda installation for running MDLite workflow has a minor bug.
- I was not able to see my personal files inside the contrib folder in the explorer interface.

1.3.3 What were the most valuable features? How would you improve them?

- The saved cluster configurations
- The cost dashboard
- The saved workflow configurations: Can be improved by adding more workflows to the marketplace.
- The workflow yaml visualization

1.3.4 What is the platform missing? How might this be made into a feature?

- The platform has many impressive features which are not intuitive to the user in the first few uses. Adding videos to the documentation can help navigation. Adding a guided tour of the screen (like how editors have by highlighting important parts of the screen) can help too. Maybe at a later stage, an AI chatbot interface to answer questions can help.

1.3.5 What questions do you have about the platform's backend after using it?

- Is there a way to shut down a few nodes after starting the cluster?

Test Part 2: Shell programming with multi-process concurrency challenges

2.1 How to run

- **Parallel:** bash testid.sh $NUM_PROCESSES $PER_PROCESS_IDS
- **Sequential:** bash testid.sh $NUM_IDS seq

2.2 Algorithm

1. Genid function
    a. Open file1 and create file descriptor
    b. Acquire lock
    c. Load current id from file2
    d. Increment current id
    e. Write the id back to file2
    f. Release lock
2. Calling the function
    a. In a for loop of n processes, create a process which runs the genid function m times where n = num_processes and m = per_process_function_runs

2.3 Performance:

The parallel execution is slow because of the parallelization overhead of running the genid function many times (the locking makes it slower than sequential run)

Number of ids generated	Parallel execution time (in ms)	Serial execution time (in ms)	Ratio (=parallel/seq)
1000	4238	42	100
2000	9437	75	125
3000	14757	109	135
4000	19361	142	137
5000	26188	184	142
6000	29998	208	144
7000	35938	239	150
8000	40261	279	144
9000	45702	326	140
10000	52358	360	146

2.4 Test method:

Generate a sequential array and compare from ids.txt.

2.5 Correctness:

Since each process contests for the same lock, and each function run increments the id written by the previous process run, the ids are serial no matter what order the processes run in.