Parallel Python

Parallel Python using mpi4py on Idun/epic cluster.

- Intel/2019a on Idun (mpi4py is installed)
- Tensorflow for GPUs and OpenMPI (mpi4py is installed) on EPIC

Intel/2019a on Idun (mpi4py is installed)

Modules

- module load intel/2019a
- module load SciPy-bundle/2019.03

(Python 3.7.2 is loaded and mpi4py is installed)

Example code (testmpi.py)

```python
from mpi4py import MPI
comm=MPI.COMM_WORLD
myrank=comm.Get_rank()
ranks=comm.Get_size()
print("hello world from rank {}, of {} ranks".format(myrank,ranks))
```

Job script (job.sh)

```bash
#!/bin/bash
#SBATCH -J job                  # sensible name for the job
#SBATCH -N 2                    # Allocate 2 nodes for the job
#SBATCH --ntasks-per-node=1    # 1 task per node
#SBATCH -t 00:10:00             # Upper time limit for the job
#SBATCH -p WORKQ
module load intel/2019a
module load SciPy-bundle/2019.03
mpirun python3 testmpi.py
```

(Tensorflow for GPUs and OpenMPI (mpi4py is installed) on EPIC)

Modules:

- module load fosscuda/2019a
- module load TensorFlow/1.13.1-Python-3.7.2

Example code (testmpi.py)

```python
from mpi4py import MPI
import tensorflow as tf
comm=MPI.COMM_WORLD
myrank=comm.Get_rank()
ranks=comm.Get_size()
print("hello world from rank {}, of {} ranks".format(myrank,ranks))
if tf.device("/gpu:0"):
    print("GPU implementet")
```

Job script (job.sh):

```bash
```

(Tensorflow for GPUs and OpenMPI (mpi4py is installed) on EPIC)
#!/bin/bash

#SBATCH --job-name job       # Sensible name for the job
#SBATCH --nodes=2             # Allocate 2 nodes for the job
#SBATCH --ntasks-per-node=1  # 1 task per node
#SBATCH --gres=gpu:1
#SBATCH --time=00:10:00       # Upper time limit for the job
#SBATCH -p EPICALL

module load fosscuda/2019a
module load TensorFlow/1.13.1-Python-3.7.2

time mpirun python3 testmpi.py

(Python 3.7.2 is loaded and mpi4py is installed)

Start

sbatch job.sh