Class on 06-02

Pickle for Memory Serialization

CPickle is also there for Python 2

In python 3, its integrated in Pickle

```
from mpi4py import MPI
comm = MPI.COMM_WORLD
comm.send(a)
comm.Send(a)
```

This creates a bufferable object in python

Send is used to transfer bufferable objects in python whereas send is used for general python objects

Home Work for today is to code the Tree Structure implementation of Addition and compare the Wall time of all the processes and report which of them is better and possibly why

Send and Receive Arbitary (n) size of Data !

Using Status Command for Sharing objects:

Using Probe command

```
from mpi4py import MPI
import numpy as np
import pickle
comm = MPI.COMM_WORLD
rank = comm.Get_rank()
size = comm.Get_size()

if rank == 0:
    data = rank*np.ones(3,dtype= np.float64)
    print data
    comm.Send([data,MPI.DOUBLE],dest=1,tag=1)
if rank ==1:
    info = MPI.Status()
    comm.Probe(MPI.ANY_SOURCE,MPI.ANY_TAG,info)
    source = info.Get_source()
    tag = info.Get_tag()
    count = info.Get_elements(MPI.DOUBLE)
    size = info.Get_count()
    print('On {} Source - {}, tag - {}, count -{}, size-
{}').format(rank,source,tag,count,size)
```

Probe Command is used to obtain the details of the data being Sent to the process.

Using Receive command

```
from mpi4py import MPI
```

```
import numpy as np
import pickle
comm = MPI.COMM_WORLD
rank = comm.Get_rank()
size = comm.Get_size()

data = rank*np.ones(3,dtype= np.float64) # Note the Variable needs to be
declared in this case
print data
if rank == 0:
    comm.Send([data,MPI.DOUBLE],dest=1,tag=1)
if rank ==1:
    info = MPI.Status()
    comm.Recv(data,MPI.ANY_SOURCE,MPI.ANY_TAG,info)
    source = info.Get_source()
    tag = info.Get_tag()
    count = info.Get_elements(MPI.DOUBLE)
    size = info.Get_count()
    print('On {} Source - {}, tag - {}, count -{}, size-{}, data-
{}').format(rank,source,tag,count,size,data)
```