EE 5243

Programming Techniques in Cloud

Counting Stars Mini Project

Submitted By,

S M Azharul Karim

Banner- @01519097

Python Code:

#!/usr/bin/env python

####################################################################################

# Image Processing example

# Author : S M Azharul Karim

# This program requires image file to be downloaded.

# wget http://www.spacetelescope.org/static/archives/images/publicationtiff40k/heic1502a.tif

#

####################################################################################

import sys

import numpy as np

import time

from mpi4py import MPI

import cv2

comm = MPI.COMM\_WORLD

rank = comm.Get\_rank()

size=comm.Get\_size()

img=np.array([0])

local\_img=np.array([0])

height=np.zeros(1)

width=np.zeros(1)

count=0

star\_count\_node=0

star\_count=np.array([0])

if rank == 0:

tic=MPI.Wtime()

t1 = time.time()

img = cv2.imread('heic1502a.tif',0)

t2 = time.time()

if img is None:

print "image hasn't loaded correctly, aborting!"

sys.exit("errors!!")

height[0],width[0]=img.shape

print "height=",height[0]," width=",width[0]

print " Time taken to open and read the image is : %r sec " %(t2-t1)

# (12788, 40000) Size of the image

# each of this part will be sent to the other nodes for processing

print " sending the parts of image to different nodes "

img=np.array(img)

print "shape of loaded image file= ",img.shape

img=np.reshape(img,(1,height[0]\*width[0]))

# transmitting the sizes to each node to create local matrix

comm.Bcast(height,root=0)

comm.Bcast(width,root=0)

#local matrix which will store local star count

local\_img=np.uint8(np.zeros((1,height[0]\*width[0]/size)))

comm.Scatter(img,local\_img,root=0)

local\_img=np.reshape(local\_img,(height[0],width[0]/size))

print "I am ",rank, " and I've got shape= ",local\_img.shape

#counting the stars in each node

count=cv2.adaptiveThreshold(local\_img,255,cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C,cv2.THRESH\_BINARY,59,0)

star\_count\_node=((200<count)).sum()

print "star count in",rank,"= ",star\_count\_node

#collecting star count from all nodes

#star\_count=comm.gather(star\_count\_node,root=0)

comm.Reduce(star\_count\_node,star\_count,op=MPI.SUM,root=0)

#summing all the star counts together in root node

if rank==0:

print "total star count= ",star\_count

toc=MPI.Wtime()-tic

print "time required to count the stars",toc

Output:

Note: The code is generalized(almost). It will work for almost any number of processes. However, The number of processes must evenly divide 40000. Otherwise the code won't work.

smazharulkarim@ubuntu:~/project$ mpiexec -n 10 python ~/Desktop/project.py

Time taken to open and read the image is : 93.72187632898237 sec

sending the parts of image to different nodes

shape of loaded image file= (12788, 40000)

I am 1 and I've got shape= (12788, 4000)

I am 2 and I've got shape= (12788, 4000)

I am 3 and I've got shape= (12788, 4000)

I am 4 and I've got shape= (12788, 4000)

I am 5 and I've got shape= (12788, 4000)

I am 6 and I've got shape= (12788, 4000)

I am 7 and I've got shape= (12788, 4000)

I am 8 and I've got shape= (12788, 4000)

I am 9 and I've got shape= (12788, 4000)

I am 0 and I've got shape= (12788, 4000)

star count in 4 = 23818541

star count in 5 = 24985047

star count in 6 = 25355738

star count in 2 = 19338229

star count in 3 = 21278962

star count in 1 = 17599988

star count in 8 = 24728188

star count in 9 = 23203546

star count in 0 = 15410354

star count in 7 = 25304505

total star count= [221023098]

time required to count the stars 108.763458087