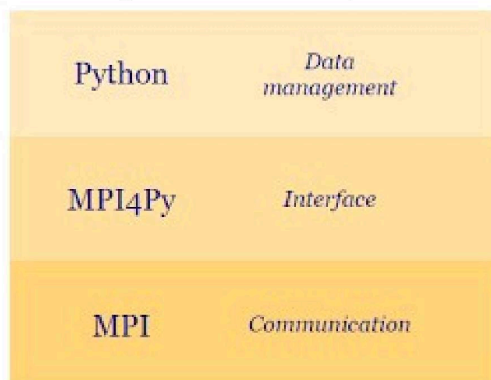


AIM:

Implement MPI communication for efficient (speedup) task scheduling between master and slaves for distributed environment (for RSA Algorithm)

THEORY:

mpi4py is a Python module that provides bindings for MPI (Message Passing Interface), a standard for parallel and distributed applications using MPI, providing Python developers with the tools to harness the power of parallel computing for solving complex computational problems efficiently.



Message Passing Interface (MPI): MPI is a standard protocol used for communication and coordination between multiple processes running on separate processors in a parallel computing environment. It

allows processes to exchange data and synchronize their activities to solve large-scale computational problems efficiently.

mpi4py Module: mpi4py is a Python module that allows Python programs to interface with MPI implementations, enabling the development of parallel and distributed applications in Python. It provides a Pythonic interface to the MPI standard, allowing Python developers to utilize MPI features such as process creation, communication, and synchronization.

Key Features of mpi4py:

Process Creation: mpi4py enables the creation and management of parallel processes, allowing Python programs to run in parallel across multiple processors or nodes in a cluster.

Communication: mpi4py provides functions for sending and receiving messages between processes, facilitating data exchange and coordination in parallel applications.

Synchronization: mpi4py supports synchronization primitives such as barriers, which allow processes to coordinate and synchronize their execution.

Collective Operations: mpi4py includes functions for performing collective operations such as broadcast, scatter, gather, and reduce, which enable efficient parallel computation and data distribution.

Use Cases: mpi4py is commonly used in scientific computing, high-performance computing (HPC), and parallel processing applications, where large-scale numerical simulations, data analysis, and other compute-intensive tasks are performed across multiple processors or nodes in a distributed computing environment.

STEPS:

Open Oracle Virtualbox, Open terminal, Run the cmds:

Install python using the following cmds:

→ sudo apt install python3

```
asrani_sanjana@asranisanjana-VirtualBox:~$ sudo apt install python3
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3 is already the newest version (3.8.2-0ubuntu2).
python3 set to manually installed.
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi
  libgstreamer-plugins-bad1.0-0 libva-wayland2
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 512 not upgraded.
asrani_sanjana@asranisanjana-VirtualBox:~$ python

Command 'python' not found, did you mean:

  command 'python3' from deb python3
  command 'python' from deb python-is-python3
```

→ sudo apt install software-properties-common

```
asrani_sanjana@asranisanjana-VirtualBox:~$ sudo apt install software-properties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

→ sudo add-apt-repository ppa:deadsnakes/ppa

```
asrani_sanjana@asranisanjana-VirtualBox:~$ sudo add-apt-repository ppa:deadsnakes/ppa
This PPA contains more recent Python versions packaged for Ubuntu.

Disclaimer: there's no guarantee of timely updates in case of security problems
or other issues. If you want to use them in a security-or-otherwise-critical env
ironment (say, on a production server), you do so at your own risk.
```

→ sudo apt update

```
asrani_sanjana@asranisanjana-VirtualBox:~$ sudo apt update
Hit:1 http://packages.microsoft.com/repos/code stable InRelease
Hit:2 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:3 http://in.archive.ubuntu.com/ubuntu focal InRelease
Hit:4 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:5 http://ppa.launchpad.net/deadsnakes/ppa/ubuntu focal InRelease
Hit:6 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
508 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

→ sudo apt install python3.12

```
asrani_sanjana@asranisanjana-VirtualBox:~$ sudo apt install python3.12
Reading package lists... Done
Building dependency tree
```

Check if installed:

→ python3

```
asrani_sanjana@asranisanjana-VirtualBox:~$ python3
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
```

Check for the python version installed:

→ python --version

```
asrani_sanjana@asranisanjana-VirtualBox:~$ python --version

Command 'python' not found, did you mean:

  command 'python3' from deb python3
  command 'python' from deb python-is-python3

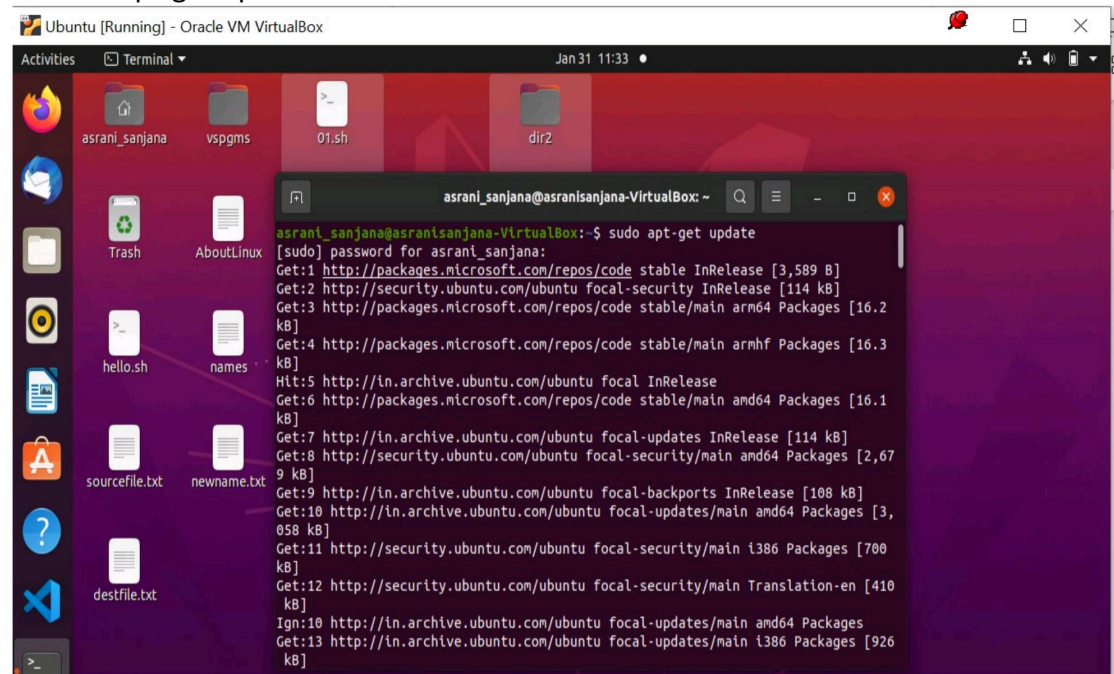
asrani_sanjana@asranisanjana-VirtualBox:~$
```


Or just run this cmd:

→ sudo apt install python3-pip

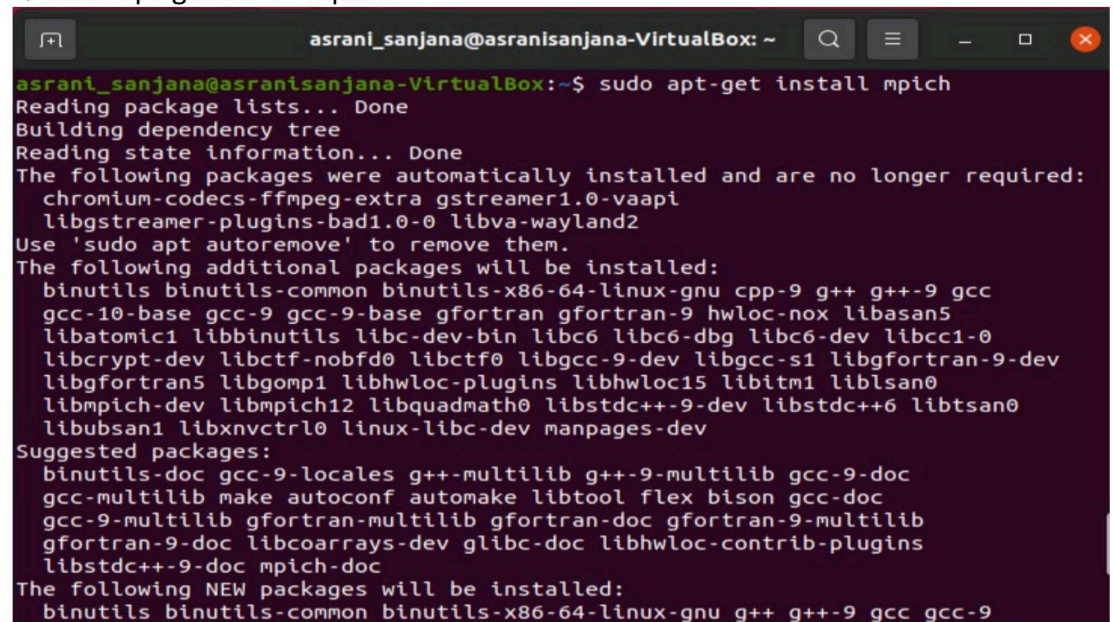
```
asrani_sanjana@asranisanjana-VirtualBox:~$ sudo apt install python3-pip
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
 chromium-codecs-ffmpeg-extra gir1.2-goa-1.0 gstreamer1.0-vaapi
```

→ sudo apt-get update



```
asrani_sanjana@asranisanjana-VirtualBox:~$ sudo apt-get update
[sudo] password for asrani_sanjana:
Get:1 http://packages.microsoft.com/repos/code stable InRelease [3,589 B]
Get:2 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:3 http://packages.microsoft.com/repos/code stable/main arm64 Packages [16.2 kB]
Get:4 http://packages.microsoft.com/repos/code stable/main armhf Packages [16.3 kB]
Hit:5 http://in.archive.ubuntu.com/ubuntu focal InRelease
Get:6 http://packages.microsoft.com/repos/code stable/main amd64 Packages [16.1 kB]
Get:7 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:8 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [2,679 kB]
Get:9 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [3,058 kB]
Get:11 http://security.ubuntu.com/ubuntu focal-security/main i386 Packages [700 kB]
Get:12 http://security.ubuntu.com/ubuntu focal-security/main Translation-en [410 kB]
Ign:10 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages
Get:13 http://in.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [926 kB]
```

→ sudo apt-get install mpich



```
asrani_sanjana@asranisanjana-VirtualBox:~$ sudo apt-get install mpich
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
 chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi
 libgstreamer-plugins-bad1.0-0 libva-wayland2
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
 binutils binutils-common binutils-x86-64-linux-gnu cpp-9 g++ g++-9 gcc
 gcc-10-base gcc-9 gcc-9-base gfortran gfortran-9 hwloc-nox libasan5
 libatomic1 libbinutils libc-dev-bin libc6 libc6-dbg libc6-dev libcc1-0
 libcrypt-dev libctf-nobfd0 libctf0 libgcc-9-dev libgcc-s1 libgfortran-9-dev
 libgfortran5 libgomp1 libhwloc-plugins libhwloc15 libitm1 liblsan0
 libmpich-dev libmpich12 libquadmath0 libstdc++-9-dev libstdc++6 libtsan0
 libubsan1 libxnvctrl0 linux-libc-dev manpages-dev
Suggested packages:
 binutils-doc gcc-9-locales g++-multilib g++-9-multilib gcc-9-doc
 gcc-multilib make autoconf automake libtool flex bison gcc-doc
 gcc-9-multilib gfortran-multilib gfortran-doc gfortran-9-multilib
 gfortran-9-doc libcoarrays-dev glibc-doc libhwloc-contrib-plugins
 libstdc++-9-doc mpich-doc
The following NEW packages will be installed:
 binutils binutils-common binutils-x86-64-linux-gnu g++ g++-9 gcc gcc-9
```

Install the module:

→ pip install mpi4py

```
asrani_sanjana@asranisanjana-VirtualBox:~$ pip install mpi4py
Collecting mpi4py
  Downloading mpi4py-3.1.5.tar.gz (2.5 MB)
    | 2.5 MB 98 kB/s
Installing build dependencies ... \
```

Working of code:

The code distributes the work among processes using MPI. Each process computes the square of a subset of numbers and performs RSA encryption and decryption on a fixed message. The results are collected in the results list.

Running the following code:

```
from mpi4py import MPI
import time

def compute_square(number):
    return number ** 2

def main():
    numbers = [2, 4, 6]
    results = []
    start_time = time.time()
    for number in numbers:
        result = compute_square(number)
        results.append(result)
    end_time = time.time()
    print("Results:", results)
    print("Execution time:", end_time - start_time, "seconds")

if __name__ == "__main__":
    main()
```

Cmd:

→ `mpiexec -n 4 python3 sanjdc03.py`

```
asrani_sanjana@asranisanjana-VirtualBox:~$ mpiexec -n 4 python3 sanjdc03.py
Results: [4, 16, 36]
Execution time: 2.86102294921875e-06 seconds
Results: [4, 16, 36]
Execution time: 3.814697265625e-06 seconds
Results: [4, 16, 36]
Execution time: 3.814697265625e-06 seconds
Results: [4, 16, 36]
Execution time: 3.5762786865234375e-06 seconds
asrani_sanjana@asranisanjana-VirtualBox:~$
```

Modifying the implementation for a different purpose i.e. performing RSA:

Code:

```
from mpi4py import MPI
import time
import math

def compute_square(number):
    return number ** 2

def gcd(a, h):
    temp = 0
    while 1:
        temp = a % h
        if temp == 0:
            return h
        a = h
        h = temp

def rsa_encrypt(msg, e, n):
    return pow(msg, e, n)

def rsa_decrypt(c, d, n):
    return pow(c, d, n)

def main():
    comm = MPI.COMM_WORLD

    rank = comm.Get_rank()
    size = comm.Get_size()

    numbers = [2, 4, 6]
    results = []

    p = 53
    q = 59
    msg = 89
    n = p * q
    e = 3
    z = (p - 1) * (q - 1)

    while e < z:
        if gcd(e, z) == 1:
            break
        else:
            e = e + 1

    k = 2
    d = pow(e, -1, z)

    if rank == 0:
        start_time = time.time()
```

```

for i in range(rank, len(numbers), size):
    number = numbers[i]
    result = compute_square(number)
    encrypted_result = rsa_encrypt(result, e, n)
    decrypted_result = rsa_decrypt(encrypted_result, d, n)
    results.append(decrypted_result)

if rank == 0:
    end_time = time.time()
    print("Given data: p=",p," , q=",q, " , e=",e," msg=",msg, "\nn=n=p*q=",p,"*",q,"=",n, " , phi(n)= ",z)

print("d=M.I. of",e,"in domain ",z,"=",d)
# Encryption c = (msg ^ e) % n
c = pow(msg, e, n)
print("Encrypted data = ", c)
# Decryption m = (c ^ d) % n
m = pow(c, d, n)
print("Original Message Sent = ", m)
print("Execution time:", end_time - start_time,
      "seconds")

if __name__ == "__main__":
    main()

```

```

asranti_sanjana@asranisanjana-VirtualBox: ~/Desktop
rsa_dc03.py
~/Desktop

asranti_sanjana@asranisanjana-VirtualBox:~/Desktop$ mplexec python3 rsa_dc03.py
Given data: p= 53 , q= 59 , e= 3 msg= 89
n=n=p*q= 53 * 59 = 3127 , phi(n)= 3016
d=M.I. of 3 in domain 3016 = 2011
Encrypted data = 1394
Original Message Sent = 89
Execution time: 8.821487426757812e-06 seconds
asranti_sanjana@asranisanjana-VirtualBox:~/Desktop$

```

Adjust the number of MPI processes based on your system configuration. See the time difference between the execution of the 2 codes.

→ `Mpiexec -n 3 python3 rsa_dc03.py`

This cmd launches the specified Python script (your_mpirsa_dc03.py) with a total of 16 MPI processes allowing for parallel execution of the code within the script across multiple processors or nodes in a distributed-memory environment.

```

asranti_sanjana@asranisanjana-VirtualBox:~/Desktop$ mpiexec -n 3 python3 rsa_dc03.py
Given data: p= 53 , q= 59 , e= 3 msg= 89
n=n=p*q= 53 * 59 = 3127 , phi(n)= 3016
d=M.I. of 3 in domain 3016 = 2011
Encrypted data = 1394
Original Message Sent = 89
Execution time: 6.4373016357421875e-06 seconds
asranti_sanjana@asranisanjana-VirtualBox:~/Desktop$ mpiexec -n 4 python3 rsa_dc03.py
Given data: p= 53 , q= 59 , e= 3 msg= 89
n=n=p*q= 53 * 59 = 3127 , phi(n)= 3016
d=M.I. of 3 in domain 3016 = 2011
Encrypted data = 1394
Original Message Sent = 89
Execution time: 5.9604644775390625e-06 seconds
asranti_sanjana@asranisanjana-VirtualBox:~/Desktop$ mpiexec -n 5 python3 rsa_dc03.py
Given data: p= 53 , q= 59 , e= 3 msg= 89
n=n=p*q= 53 * 59 = 3127 , phi(n)= 3016
d=M.I. of 3 in domain 3016 = 2011
Encrypted data = 1394
Original Message Sent = 89
Execution time: 5.7220458984375e-06 seconds
asranti_sanjana@asranisanjana-VirtualBox:~/Desktop$

```

Now run this using cuda in colab for comparison:


```

# Install mpi4py (MPI for Python)
!pip install mpi4py

# Install CUDA toolkit
!apt update
!apt install -y cuda

# Install CUDA-aware MPI support for mpi4py
!pip install mpi4py-fft

# Next, let's write the MPI-like code using multiprocessing
from mpi4py import MPI
import time
import math
from multiprocessing import Pool

def compute_square(number):
    """ return number**2 """

def gcd(a, h):
    """ temp = 0
    while 1:
        temp = a % h
        if temp == 0:
            return h
        a = h
        h = temp """

def rsa_encrypt(msg, e, n):
    """ return pow(msg, e, n) """

def rsa_decrypt(c, d, n):
    """ return pow(c, d, n) """

def main():
    """ rank = 0
    size = MPI.COMM_WORLD.Get_size()

    numbers = [2, 4, 6]
    results = []

    p = 53
    q = 59
    msg=89
    n = p*q
    e = 3
    z = (p - 1) * (q - 1)

    while e < z:
        if gcd(e, z) == 1:
            break
        else:
            e = e + 1

    k = 2
    d = pow(e, -1, z)

    if rank == 0:
        start_time = time.time()

    with Pool(processes=size) as pool:
        results = pool.map(compute_square, numbers)

    if rank == 0:
        end_time = time.time()
        print("Given data: p=", p, ", q=", q, ", e=", e, " msg=", msg, "\nn=n=p*q=", p, "*", q, "=", n, ", phi(n)= ", z)
        print("d=M.I. of", e, "in domain ", z, "=", d)
        """ # Encryption c = (msg ^ e) % n
        c = pow(msg, e, n)
        print("Encrypted data = ", c)
        """ # Decryption m = (c ^ d) % n
        m = pow(c, d, n)
        print("Original Message Sent = ", m)
        print("Execution time:", end_time - start_time, "seconds")

if __name__ == "__main__":
    main()

```



```
Running module version sanity check.
- Original module
  - No original module exists within this kernel
- Installation
  - Installing to /lib/modules/5.15.0-92-generic/updates/dkms/

depmod...
Setting up nvidia-driver-545 (545.23.08-0ubuntu1) ...
Setting up cuda-drivers-545 (545.23.08-1) ...
Setting up cuda-drivers (545.23.08-1) ...
Setting up cuda-runtime-12-3 (12.3.2-1) ...
Setting up cuda-demo-suite-12-3 (12.3.101-1) ...
Setting up cuda-12-3 (12.3.2-1) ...
Setting up cuda (12.3.2-1) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for dbus (1.12.20-2ubuntu4.1) ...
Processing triggers for fontconfig (2.13.1-4.2ubuntu5) ...
Processing triggers for hicolor-icon-theme (0.17-2) ...
Processing triggers for libc-bin (2.35-0ubuntu3.4) ...
/sbin/ldconfig.real: /usr/local/lib/libtbbbind.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbb.so.12 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_0.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc_proxy.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_5.so.3 is not a symbolic link

Collecting mpi4py-fft
  Downloading mpi4py-fft-2.0.5.tar.gz (36 kB)
  error: subprocess-exited-with-error

  × python setup.py egg_info did not run successfully.
    exit code: 1
  ──> See above for output.

  note: This error originates from a subprocess, and is likely not a problem with pip.
  Preparing metadata (setup.py) ... error
error: metadata-generation-failed

× Encountered error while generating package metadata.
  ──> See above for output.

  note: This is an issue with the package mentioned above, not pip.
  hint: See above for details.
Given data: p= 53 , q= 59 , e= 3 msg= 89
n=n*p*q= 53 * 59 = 3127 , phi(n)= 3016
d=M.I. of 3 in domain 3016 = 2011
Encrypted data = 1394
Original Message Sent = 89
Execution time: 0.026618480682373047 seconds
```