

TUGAS KELOMPOK
PEMROSESAN PARALEL



DOSEN PENGAMPU :
AHMAD HERYANTO, S.KOM, M.T.
ADI HERMANSYAH, S.KOM., M.T.

PEMBUAT (KELOMPOK 10):

ANATA RYU ILHAMI	(09011382126168)
AGUNG RIZQI RAMADHAN	(09011382126157)
NANDA APRIADI	(09011382126170)

FAKULTAS ILMU KOMPUTER
JURUSAN SISTEM KOMPUTER
UNIVERSITAS SRIWIJAYA

Laporan Praktikum Eksekusi Program Numerik Python Menggunakan MPI & Numpy Secara Paralel di Ubuntu

Kode yang dibuat dalam mengeksekusi program python tersebut :

```
from mpi4py import MPI
import numpy as np
import time

# Inisialisasi MPI
comm = MPI.COMM_WORLD
rank = comm.Get_rank()
size = comm.Get_size()

# Panjang array
array_length = 100

# Bagi pekerjaan sesuai jumlah proses
local_array_length = array_length // size
local_array = np.random.rand(local_array_length)

# Memulai pengukuran waktu
start_time = time.time()

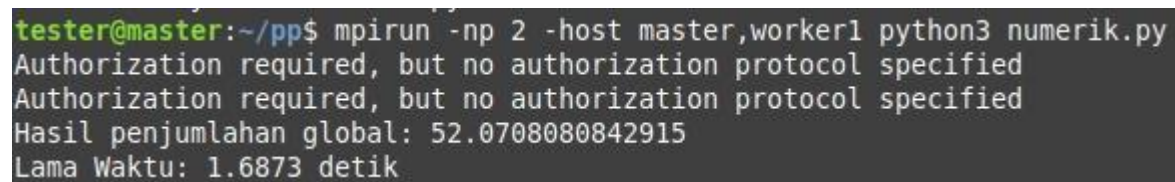
# Menjumlahkan elemen-elemen lokal
local_sum = np.sum(local_array)

# Menggunakan operasi reduksi untuk menghitung jumlah global
global_sum = comm.reduce(local_sum, op=MPI.SUM, root=0)

# Menghentikan pengukuran waktu
end_time = time.time()

# Proses 0 mencetak hasil dan waktu eksekusi
if rank == 0:
    print("Hasil penjumlahan global:", global_sum)
    print("Waktu eksekusi: {:.4f} detik".format(end_time - start_time))
```

Setelah itu eksekusi kode tersebut kedalam mpi4python seperti dibawah ini.



```
tester@master:~/pp$ mpirun -np 2 -host master,worker1 python3 numerik.py
Authorization required, but no authorization protocol specified
Authorization required, but no authorization protocol specified
Hasil penjumlahan global: 52.0708080842915
Lama Waktu: 1.6873 detik
```

Penjelasan :

Program diatas merupakan program sederhana untuk melakukan perhitungan numerik pada suatu array dengan komputasi paralel

Program yang Anda berikan adalah program Python yang menggunakan MPI (Message Passing Interface) untuk melakukan komputasi paralel dan menghitung jumlah elemen dalam sebuah array numerik. Di bawah ini adalah penjelasan langkah demi langkah mengenai program tersebut:

1. Inisialisasi MPI
 - Program dimulai dengan mengimpor modul `'MPI'` dari `'mpi4py'` untuk mengakses fitur MPI.
 - Kemudian, program menginisialisasi lingkungan MPI dengan `'MPI.COMM_WORLD'`.
 - Variabel `'rank'` menyimpan nomor rank (identifikasi) dari setiap proses dalam komunikasi MPI.
 - Variabel `'size'` menyimpan jumlah total proses dalam komunikasi MPI.
2. Panjang Array
 - Variabel `'array_length'` menunjukkan panjang array numerik yang akan digunakan dalam perhitungan. Dalam kasus ini, panjang array adalah 100.
3. Pembagian Pekerjaan:
 - Panjang array dibagi menjadi bagian-bagian yang sesuai dengan jumlah proses. Setiap proses akan bekerja pada bagian array yang berbeda.
 - `'local_array_length'` menyimpan panjang bagian array yang akan dihitung oleh setiap proses.
 - `'local_array'` adalah array lokal yang diisi dengan bilangan acak dengan panjang `'local_array_length'` menggunakan NumPy.
4. Pengukuran Waktu Awal:
 - Sebelum komputasi dimulai, waktu saat ini diukur dengan `'time.time()'` dan disimpan dalam `'start_time'`.
5. Perhitungan Lokal:
 - Setiap proses menghitung jumlah elemen dalam `'local_array'` menggunakan `'np.sum()'` dan hasilnya disimpan dalam `'local_sum'`.
6. Operasi Reduksi:
 - Setelah perhitungan lokal selesai, operasi reduksi digunakan untuk menggabungkan hasil perhitungan dari semua proses dan menghitung jumlah global.
 - `'comm.reduce(local_sum, op=MPI.SUM, root=0)'` mengumpulkan hasil `'local_sum'` dari semua proses dan menyimpannya di `'global_sum'` dengan operasi penjumlahan (`'MPI.SUM'`).
 - Hasil akhir dari jumlah global akan tersedia di proses dengan rank 0.
7. Pengukuran Waktu Akhir:
 - Setelah komputasi selesai, waktu saat ini diukur kembali dengan `'time.time()'` dan disimpan dalam `'end_time'`.
8. Pencetakan Hasil dan Waktu Eksekusi:
 - Proses dengan rank 0 (biasanya proses utama) mencetak hasil penjumlahan global dari semua proses dan mencetak waktu eksekusi yang dihitung dari selisih `'end_time - start_time'`.

Program ini mengilustrasikan cara menggunakan MPI untuk melakukan komputasi paralel sederhana dan menghitung jumlah elemen dalam array numerik. Itu juga mencakup pengukuran waktu eksekusi untuk memantau kinerja komputasi.