Assignment-3

Name: K.Amulya

H.NO:2303A51242

Batch:05

Assignment 3: Load Balancing with Irregular Workloads

Scenario:

An image processing pipeline processes images with different resolutions, leading to uneven computation time.

Tasks

1. Simulate variable workload per iteration.
2. Parallelize using p range.
3. Observe execution time variation.
4. Discuss limitations of scheduling in Python.

Code:

```
import time
import numpy as np
from numba import njit, prange
image_sizes = np.random.randint(500, 5000, size=1000)
@njit
def process_image(size):
    total = 0.0
    for i in range(size):
        total += np.sqrt(i)
    return total
def serial_execution(sizes):
    result = 0.0

    start = time.time()

    for s in sizes:
        result += process_image(s)

    end = time.time()

    print("Serial Execution Time:", end - start)
    return result
@njit(parallel=True)
def parallel_execution(sizes):

    result = 0.0

    for i in prange(len(sizes)):
        result += process_image(sizes[i])

    return result
```

```python
if __name__ == "__main__":
    serial_execution(image_sizes)
    start = time.time()
    parallel_execution(image_sizes)
    end = time.time()
    print("Parallel Execution Time:", end - start)
    print("\nExecution Time Variation:")
    for i in range(5):
        start = time.time()
        parallel_execution(image_sizes)
        print("Run", i+1, ":", time.time() - start)
```

OUTPUT:

```
Serial Execution Time: 3.9777896404266357
Parallel Execution Time: 2.0336170196533203

Execution Time Variation:
Run 1 : 0.008420705795288086
Run 2 : 0.0066831111907958984
Run 3 : 0.008542776107788086
Run 4 : 0.006478786468505859
Run 5 : 0.006387948989868164
```

Observation:

- Parallel program runs faster than the serial program.
- Different image sizes cause unequal work for threads.
- Some threads become idle while others are still working.
- Python has limited scheduling, so performance is not always best.