Parallel Computing (BCS702)

Program - 4

Write an openMP program to find the prime numbers from 1 to n employing parallel for directive. Record both serial and parallel execution times.

```
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>
#include <math.h>
int is_prime(int num) {
    if (num <= 1) return 0;
    if (num == 2) return 1;
    if (num % 2 == 0) return 0;
    for (int i = 3; i <= sqrt(num); i += 2) {
        if (num % i == 0) return 0;
    }
    return 1;
}

int main() {
    int n;
    printf("Enter the upper limit (n) to find prime numbers: ");
    scanf("%d", &n);

    if (n < 2) {
        printf("There are no prime numbers up to %d.\n", n);
        return 0;
    }

    printf("\nFinding prime numbers from 1 to %d...\n", n);

    double start_time = omp_get_wtime();
    int sequential_prime_count = 0;
    for (int i = 1; i <= n; i++) {
        if (is_prime(i)) sequential_prime_count++;
    }
    double time_seq = omp_get_wtime() - start_time;
    printf("\nSequential: Found %d primes in %f seconds\n", sequential_prime_count, time_seq);

    start_time = omp_get_wtime();
```

```
    int parallel_prime_count = 0;
    #pragma omp parallel for reduction(+:parallel_prime_count) schedule(dynamic)
    for (int i = 1; i <= n; i++) {
        if (is_prime(i)) parallel_prime_count++;
    }
    double time_par = omp_get_wtime() - start_time;
    printf("Parallel:   Found %d primes in %f seconds\n", parallel_prime_count, time_par);

    if (time_par > 0 && time_seq > 0) {
        printf("\nSpeedup: %.2fx\n", time_seq / time_par);
    }

    return 0;
}
```

Output :

Enter the upper limit (n) to find prime numbers: 200000

Finding prime numbers from 1 to 200000...

Sequential: Found 17984 primes in 0.035000 seconds
Parallel: Found 17984 primes in 0.015000 seconds

Speedup: 2.33x