

CSE4001 - Parallel and Distributed Computing

Lab 21+22

Digital Assignment- 3

Submitted by: Alokam Nikhitha

Reg No:19BCE2555

QUESTION:

Write a C program to perform parallel matrix multiplication using OpenMP. You should first create three matrices A, B, and C then initialize A and B to some values of your choice. In your code, try to improve the performance by (re)using the same set of threads for initializing A and B and for calculating C.

CODE:

```
#include <stdio.h>

#include <omp.h>

#define NRA 3
#define NCA 2
#define NCB 2

int A[NRA][NCA];
int B[NCA][NCB];
int C[NRA][NCB];

int main() {
    omp_set_num_threads(5);
    int i,j,k;
    for(i=0; i<NRA;i++) {
        for(int j=0;j<NCA;j++) {
            A[i][j] = i+j;
        }
    }
    for(i=0; i<NCA;i++) {
```

```

        for(int j=0;j<NCB;j++) {
            B[i][j] = (i+1)*(j+1);
        }
    }

#pragma omp parallel for private(i,j,k) shared(A,B,C)
for(int i=0; i<NRA; i++) {
    for(int j=0; j<NCB; j++) {
        for(int k=0; k<NCA; k++)
            C[i][j] += A[i][k]*B[k][j];
    }
}

printf("\nMatrix A:\n");
for(int i=0;i<NRA;i++) {
    for(int j=0;j<NCA;j++)
        printf("%d\t",A[i][j]);
    printf("\n");
}

printf("\nMatrix B:\n");
for(int i=0;i<NCA;i++) {
    for(int j=0;j<NCB;j++)
        printf("%d\t",B[i][j]);
    printf("\n");
}

```

```

printf("\nResult:\n");

for(int i=0;i<NRA;i++) {

for(int j=0;j<NCB;j++)

    printf("%d\t",C[i][j]);

printf("\n");

}

return 0;

}

```

CODE SNIPPETS:

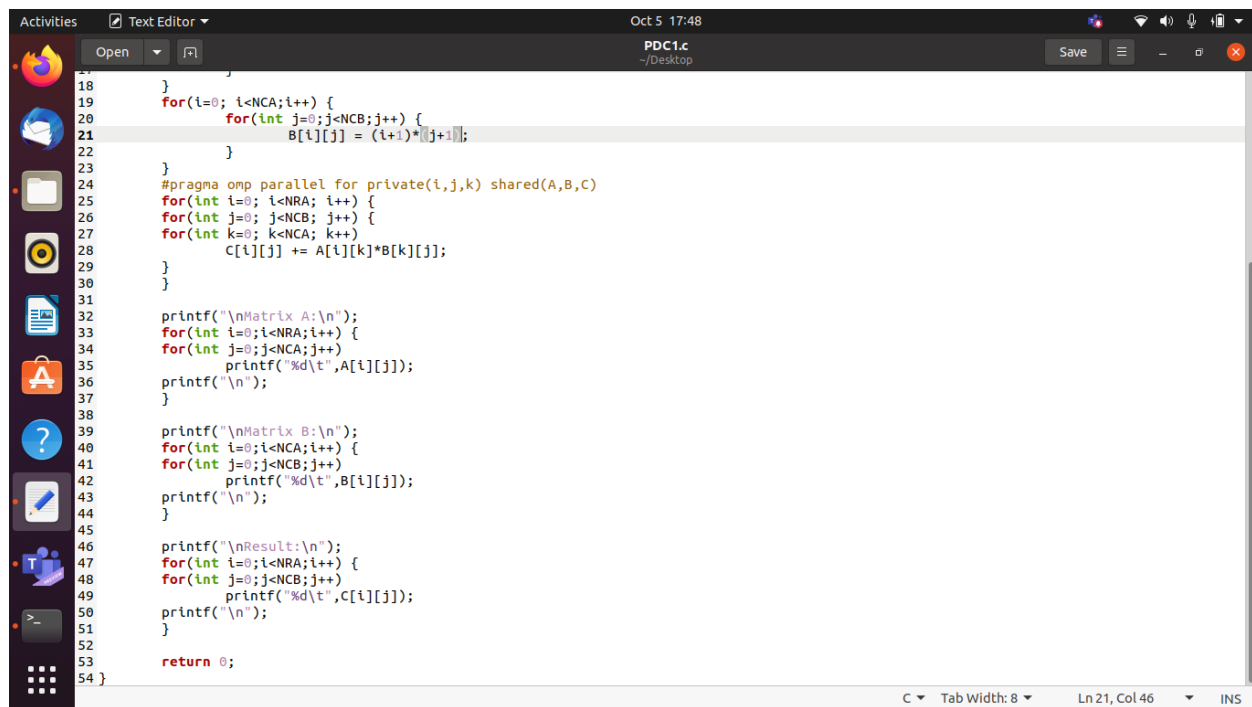


The screenshot shows a Linux desktop with a text editor window titled 'PDC1.c' at the path '~/Desktop'. The code is a C program for matrix multiplication, utilizing OpenMP for parallelization. It defines three matrices: A (NRA x NCA), B (NCA x NCB), and C (NRA x NCB). The main function sets the number of threads to 5 and performs three nested loops: a sequential loop for matrix A, a sequential loop for matrix B, and a parallel loop for matrix C using the OpenMP pragma. The status bar at the bottom indicates 'Tab Width: 8', 'Ln 32, Col 1', and 'INS' mode.

```

1 #include <stdio.h>
2 #include <omp.h>
3
4 #define NRA 3
5 #define NCA 2
6 #define NCB 2
7
8 int A[NRA][NCA];
9 int B[NCA][NCB];
10 int C[NRA][NCB];
11
12 int main() {
13     omp_set_num_threads(5);
14     int i,j,k;
15
16     for(i=0; i<NRA;i++) {
17         for(int j=0;j<NCA;j++) {
18             A[i][j] = i+j;
19         }
20     }
21
22     for(i=0; i<NCA;i++) {
23         for(int j=0;j<NCB;j++) {
24             B[i][j] = (i+1)*(j+1);
25         }
26     }
27
28     #pragma omp parallel for private(i,j,k) shared(A,B,C)
29     for(int i=0; i<NRA; i++) {
30         for(int j=0; j<NCB; j++) {
31             for(int k=0; k<NCA; k++) {
32                 C[i][j] += A[i][k]*B[k][j];
33             }
34         }
35     }
36
37     printf("\nMatrix A:\n");
38     for(int i=0;i<NRA;i++) {

```

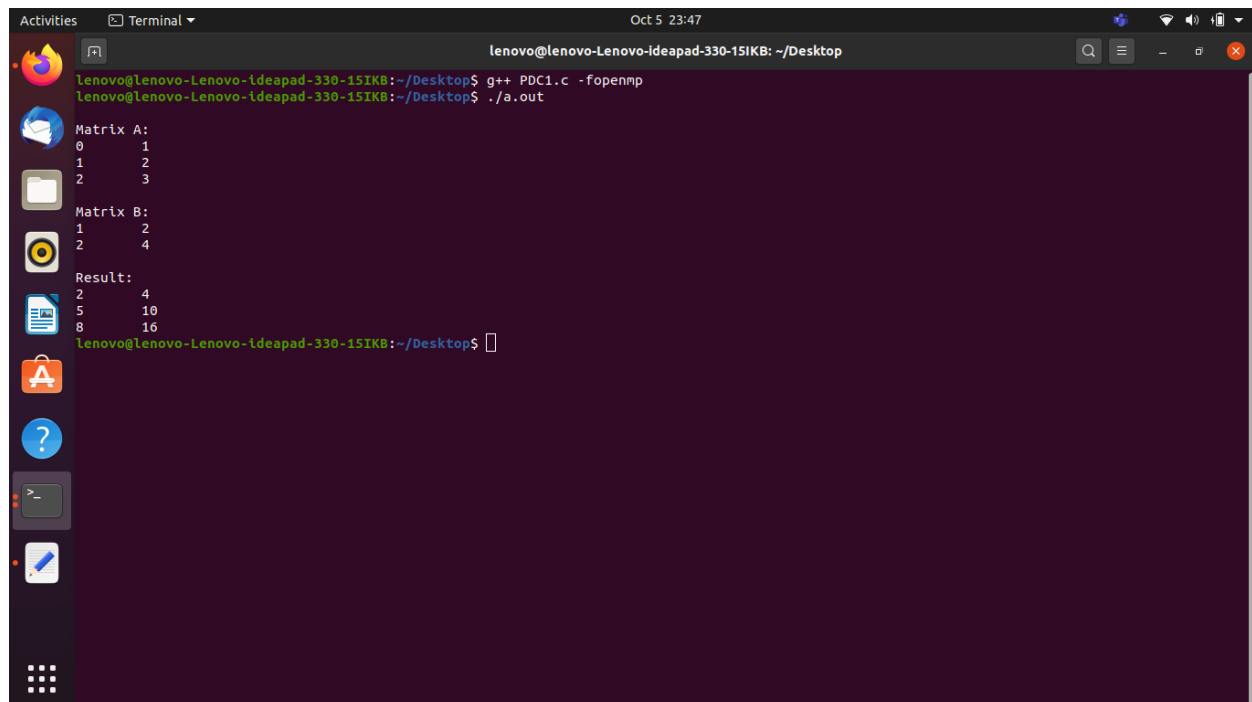


The screenshot shows a text editor window titled "PDC1.c" with the following C code:

```
18 }
19 for(i=0; i<NCA;i++) {
20     for(int j=0; j<NCB;j++) {
21         B[i][j] = (i+1)*[j+1];
22     }
23 }
24 #pragma omp parallel for private(i,j,k) shared(A,B,C)
25 for(int i=0; i<NRA; i++) {
26     for(int j=0; j<NCB; j++) {
27         for(int k=0; k<NCA; k++)
28             C[i][j] += A[i][k]*B[k][j];
29     }
30 }
31
32 printf("\nMatrix A:\n");
33 for(int i=0; i<NRA; i++) {
34     for(int j=0; j<NCA; j++)
35         printf("%d\t",A[i][j]);
36     printf("\n");
37 }
38
39 printf("\nMatrix B:\n");
40 for(int i=0; i<NCA; i++) {
41     for(int j=0; j<NCB;j++)
42         printf("%d\t",B[i][j]);
43     printf("\n");
44 }
45
46 printf("\nResult:\n");
47 for(int i=0; i<NRA; i++) {
48     for(int j=0; j<NCB;j++)
49         printf("%d\t",C[i][j]);
50     printf("\n");
51 }
52
53 return 0;
54 }
```

The status bar at the bottom indicates "C", "Tab Width: 8", "Ln 21, Col 46", and "INS".

OUTPUT:



The screenshot shows a terminal window with the following output:

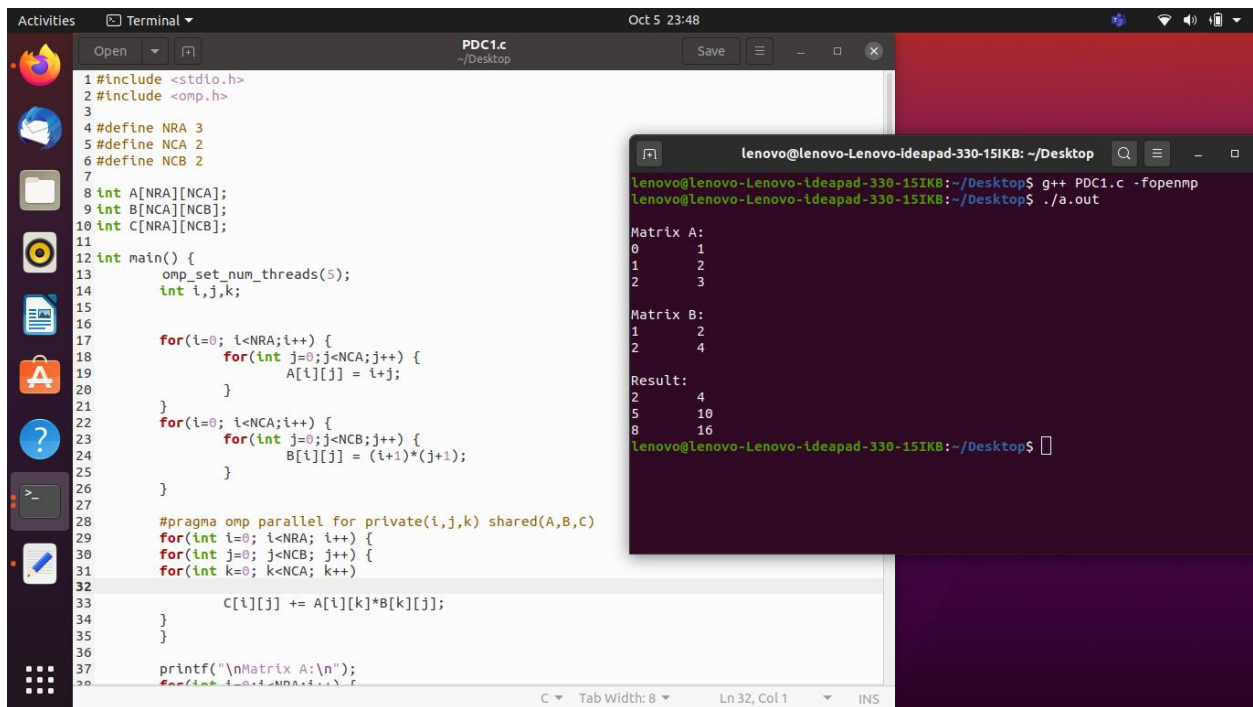
```
lenovo@lenovo-Lenovo-Ideapad-330-15IKB: ~/Desktop
lenovo@lenovo-Lenovo-Ideapad-330-15IKB:~/Desktop$ g++ PDC1.c -fopenmp
lenovo@lenovo-Lenovo-Ideapad-330-15IKB:~/Desktop$ ./a.out

Matrix A:
0 1
1 2
2 3

Matrix B:
1 2
2 4

Result:
2 4
5 10
8 16
lenovo@lenovo-Lenovo-Ideapad-330-15IKB:~/Desktop$
```

OUTPUT WITH CODE:



The screenshot shows a Linux desktop environment with a terminal window open. The terminal displays the execution of a C program named PDC1.c, which performs matrix multiplication using OpenMP for parallelization. The code defines three matrices: A (3x2), B (2x2), and C (3x2). Matrix A is initialized with values [0, 1], [1, 2], and [2, 3]. Matrix B is initialized with values [1, 2], [2, 4]. The result matrix C is calculated as the product of A and B, resulting in values [2, 4], [5, 10], and [8, 16].

```
1 #include <stdio.h>
2 #include <omp.h>
3
4 #define NRA 3
5 #define NCA 2
6 #define NCB 2
7
8 int A[NRA][NCA];
9 int B[NCA][NCB];
10 int C[NRA][NCB];
11
12 int main() {
13     omp_set_num_threads(5);
14     int i,j,k;
15
16     for(i=0; i<NRA;i++) {
17         for(int j=0;j<NCA;j++) {
18             A[i][j] = i+j;
19         }
20     }
21     for(i=0; i<NCA;i++) {
22         for(int j=0;j<NCB;j++) {
23             B[i][j] = (i+1)*(j+1);
24         }
25     }
26
27     #pragma omp parallel for private(i,j,k) shared(A,B,C)
28     for(int i=0; i<NRA; i++) {
29         for(int j=0; j<NCB; j++) {
30             for(int k=0; k<NCA; k++)
31                 C[i][j] += A[i][k]*B[k][j];
32         }
33     }
34
35     printf("\nMatrix A:\n");
36     for(int i=0; i<NRA; i++) {
37         for(int j=0; j<NCA; j++) {
38             printf("%d\t", A[i][j]);
39         }
40         printf("\n");
41     }
42
43     printf("\nMatrix B:\n");
44     for(int i=0; i<NCA; i++) {
45         for(int j=0; j<NCB; j++) {
46             printf("%d\t", B[i][j]);
47         }
48         printf("\n");
49     }
50
51     printf("\nResult:\n");
52     for(int i=0; i<NRA; i++) {
53         for(int j=0; j<NCB; j++) {
54             printf("%d\t", C[i][j]);
55         }
56         printf("\n");
57     }
58 }
```

lenovo@lenovo-Lenovo-Ideapad-330-15IKB: ~/Desktop\$ g++ PDC1.c -fopenmp
lenovo@lenovo-Lenovo-Ideapad-330-15IKB:~/Desktop\$./a.out

Matrix A:
0 1
1 2
2 3

Matrix B:
1 2
2 4

Result:
2 4
5 10
8 16
lenovo@lenovo-Lenovo-Ideapad-330-15IKB:~/Desktop\$