Lab 4 Lab Report: CprE 308

Matt McKillip

1. Introduction

Lab 4 was designed to become familiar with creating, managing, and computing using the pthreading library. Sections 2 and 3 served as review from lecture, which helped to reference during section 4.

2. Questions

2.1 Original pthread_test.c

2.1.1What is one expected output of running this program?

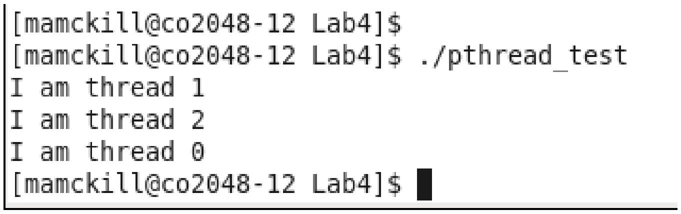
“I am thread 1
I am thread 2
I am thread 0”

2.1.2 What is the actual output of the program?

“I am thread 0”

2.2 Modified pthread_test.c

2.2.1 WHat is the output of the program



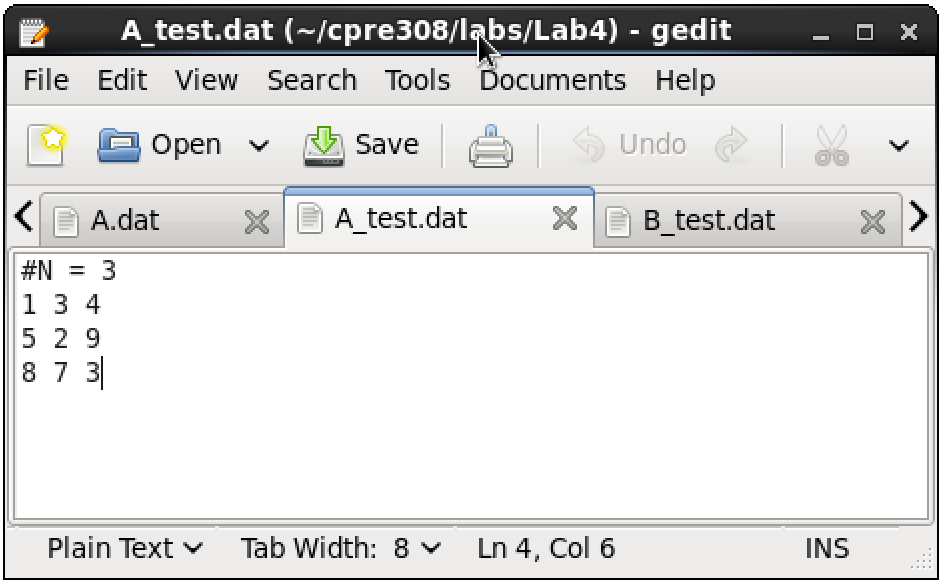
```
[mamckill@co2048-12 Lab4]$
[mamckill@co2048-12 Lab4]$ ./pthread_test
I am thread 1
I am thread 2
I am thread 0
[mamckill@co2048-12 Lab4]$ █
```

2.2.2 Does it match with the expected output of the program?

Yes, that is the order that I expected.

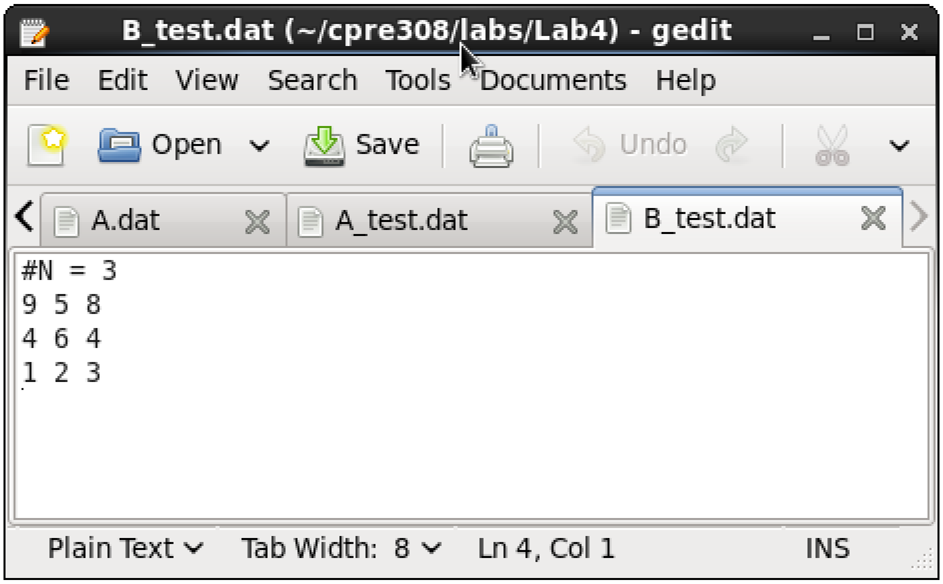
3. Results/Output

Test input matrix A


A_test.dat (~/cpre308/labs/Lab4) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
A.dat
A_test.dat
B_test.dat
Plain Text
Tab Width: 8
Ln 4, Col 6
INS

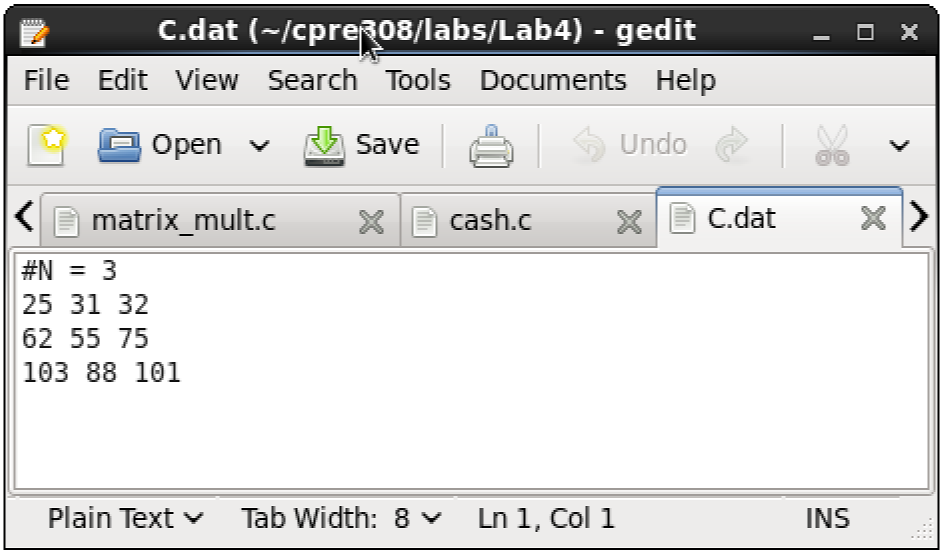
```
#N = 3
1 3 4
5 2 9
8 7 3
```

Input matrix B


B_test.dat (~/cpre308/labs/Lab4) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
A.dat
A_test.dat
B_test.dat
Plain Text
Tab Width: 8
Ln 4, Col 1
INS

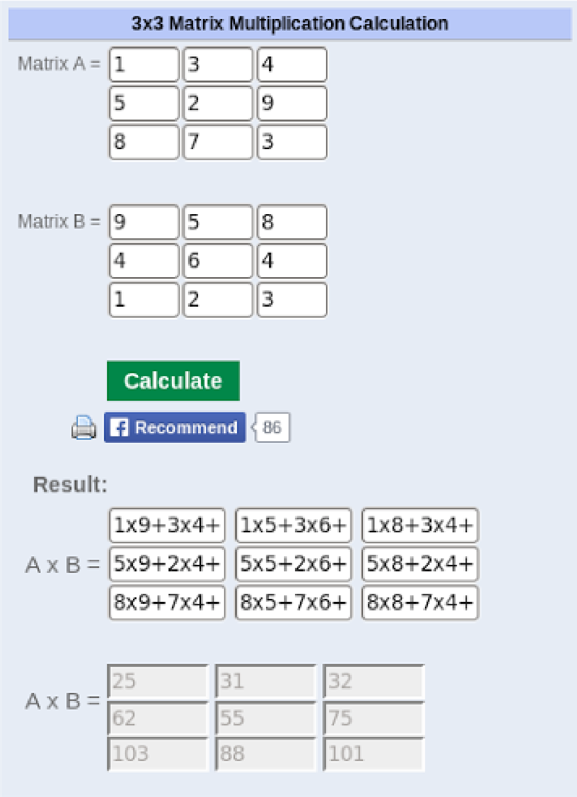
```
#N = 3
9 5 8
4 6 4
1 2 3
```

output of AxB


C.dat (~/cpre308/labs/Lab4) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
matrix_mult.c
cash.c
C.dat
Plain Text
Tab Width: 8
Ln 1, Col 1
INS

```
#N = 3
25 31 32
62 55 75
103 88 101
```

check



Matrix A =

```
1 3 4
5 2 9
8 7 3
```

Matrix B =

```
9 5 8
4 6 4
1 2 3
```

Result:

A x B =

```
1x9+3x4+ 1x5+3x6+ 1x8+3x4+
5x9+2x4+ 5x5+2x6+ 5x8+2x4+
8x9+7x4+ 8x5+7x6+ 8x8+7x4+
```

A x B =

```
25  31 32
62  55 75
103 88 101
```

4. Design Decision

I chose to implement threading by computing an entire row of the output array in a single thread. This simplified the calculation and thread inputs, but does not allow for number of threads != array size. The program can handle different matrix size by changing the “#define matrix_size 64” line. I chose this implementation because I found it to be intuitive given how matrix multiplication works. This provided easier development and testing. A more optimal solution would dynamically create the correct number of threads capping at 1024 for best performance.

5. Issues

The biggest issue I came across was deciding how to break up the calculations for threading. I originally tried to break the matrix into 2x2 matrices, but ran into trouble calculating odd sized arrays. This is why I chose to reserve a single thread for an entire row of the output matrix.

6. Conclusion

I was successful in learning and practicing pthreading. My matrix multiplication program was able to correctly compute the output matrix using pthread’s.

7. Suggestions

I thought the length of the lab was good. I would have liked more information about the input matrices. Such as; are they square matrices, are they both the same size, and what the maximum size is.