HW1: Matrix-Matrix Multiplication

Course:	Informal Parallel Programming Course for High School Students, Fall 2007
Title:	Matrix-matrix multiplication
Date Assigned:	September 25, 2007
Date Due:	October 2, 2007

1 Problem statement

Multiplying two *nxn* matrices A and B results in another *nxn* matrix C, whose elements $c_{i,j}$ satisfy $c_{i,j} = \sum_{k=1}^{n} a_{i,k}b_{k,j}$. Given two such matrices A and B, compute matrix C.

Give a serial algorithm whose time is proportional to n^3. Give a parallel algorithm whose work is proportional to n^3, and whose time is proportional to n.

2 Assignment

- Write XMTC program realizing the serial algorithm.
- Write the *pseudo-code* of the parallel algorithm using nested spawn commands. At the moment, it is not possible to implement such program because nested spawns are not fully supported.
- Write an XMTC program realizing the parallel algorithm without use of nested spawns, and implement it.

3 The program

You must name your XMTC files as follows: matmul.s.c (serial) and matmul.p.c.

3.1 Setting up the environment

The header files and the binary files can be downloaded from ∼ *swatson/xmtdata.* To get the data files, log in to your account in the class server and copy the *matmul.tgz* file from directory using the following commands:

```
$ cp ~swatson/xmtdata/matmul.tgz ~/
$ tar xzvf matmul.tgz
```

This will create the directory *matmul* with following folders: *data, src*, and *doc*. Data files are available in data directory.

3.2 Input format

You are given two matrices A and B that contains *nxn* integers and the result matrix C of size *nxn* all zeros initially.

int n	One dimension of the matrices
int N	*nxn*
int A[n][n]	The matrix A
int B[n][n]	The matrix B
int C[n][n]	The matrix C, initially zero

Table 1: Input format

3.3 Data sets

Run all your programs (serial and parallel) using the following data files. You can directly include the header file into your XMTC code with *#include* or you can include the header file with the compile option *-include*. To run the compiled program you will need to specify the binary data with *–data-file* option.

Data Set	**Header File**	**Binary file**
$n = 64, N = n^2 = 4096$	data/64/matmul.h	data/64/matmul.32b
$n = 128, N = n^2 = 16384$	data/128/matmul.h	data/128/matmul.32b
$n = 256, N = n^2 = 65536$	data/256/matmul.h	data/256/matmul.32b

Table 2: Header files

3.4 Output

Prepare and fill the following table: Create a text file named table.txt in doc. Remove any printINT and printSTR statements from your code while taking these measurements. Printf statements increase the clock and instruction count. Therefore the measurements with printf statements may not reflect the actual time and work done. Is the second parallel algorithm preferred to the first one? After having filled the table, write your arguments.

Input size	**n = 64**	**n = 128**	**n = 256**
Serial Clock Cycles			
Parallel Clock Cycles			

Table 3: Clock cycles will be written to table.txt

Explain algorithms: Create a text file named algorithms.txt in doc, and explain clearly your parallel algorithm that you run and give the pseudo-code of the algorithm using nested spawn.

3.5 Submission

The use of the make utility for submission *make submit* is required. Make sure that you have the correct files at correct locations (*src* and *doc* directories) using the make submitcheck command. Run following commands to submit the assignment:

```
$ make submitcheck
$ make submit
```

If you have any questions, please send an e-mail to Scott Watson, *swatson@umd.edu*