Victor Wu

CS 350

Design Document

MMWP – Project 4

03/20/2015

**Purpose:**

Computing a product of two matrices using processes, by forking, concurrently

**Constraints:**

* Matrices are at least 3 by 3
* Matrices does not have to be square, but must meet requirements of a valid matrix multiplication
* Allocated memory for matrices will fit within the finite machine

**Design:**

Using the existing template for Project 3 – MMWT – I just have to change the implementation from a threading program to a multi-processing program. To do so, I will need to understand that atomic operations could be faulty (data races, etc), and hence the reason for shared memory. The libraries and functions I will revolve around is fork(), shmget(), and shmat().

The use of functions will create too much overhead as there would be no need for passing in variables when they are clones of each other (hence having identical data prior to updates after fork()). Thus I will just do the matrix multiplications inside the child processes.

My design will be to split the program into two overall steps:

1. Computing the product and store it into an allocated *shared* memory and using processes to improve efficiency
2. Accesses the *shared* memory and computes the final output and store it back into memory

I will need to compute in a separate process for each multiplication, and not as a result from the resultant matrix. Take for example, Matrix A, Matrix B, and Matrix C such that: AxB = C. By applying the algorithm C:= AB + C, we can compute C.

To figure out the amount of multiplications needed: Let n and m be an integer

Where n = the amount of Columns in A

And m = the amount of Rows in B

Then A1,1B1,2+A1,2B2,1+A1,3B3,1+A1,nBm,1… = A’B’

And from the identity of matrix multiplication we can conclude:

Column of A \* Row of B \* Column of B \* Row of A = Total Multiplication

**Errors/Deviations:**

While changing the implementation of my original code for Project 3 to this two-part project, I swayed from successfully doing an ideal total multiplications amount of processes as this was carried over from my original design. Resulting in a *nxm* processes program.

While debugging my program, I realized there was a few logic errors that segment faults would come up (especially with improperly inputted matrices), possibly carried over from Project 3 as well.

The logical implementation of this project should, however, be intact.

**Improvements:**

Some improvements I would make is that I would try making the shared memory access holding a two dimensional array so that it is easier to construct and understand instead of a large linear storage for which each element in a matrix must be computed by an offset.

**Usage:**

Please refer to the –-help command for additional details.

./wu\_p4

To run in user-input mode with raw data input into stdin

./wu\_p4 [FILE1] [FILE2]

Where FILE1 is the input file, and FILE2 is the output file

**Additional Notes:**

This program should work with multiple digits for each matrix, and is flexible in design for different dimensions- though to note the fix properties of matrix multiplication.