

**CIS5930 Programming Assignment**  
**Efficient single-thread, OpenMP and MPI Matrix Multiply Routine**

## OBJECTIVES

- Practice performance tuning, OpenMP and MPI programming

## DESCRIPTION

Matrix multiply is an important function that is used in many applications. The naive implementation of this function ( $C \leftarrow A \times B$ ) is as follows:

```
for (i=0; i<size; i++)
  for (j=0; j < size; j++)
    for (k=0; k<size; k++)
      C[i][j] = C[i][j] + A[i][k] * B[k][j];
```

In this project, you will first implement a more efficient single threaded version of the routine. You will then further increase the performance by using OpenMP and MPI. The following requirement must be met:

- Your routines must produce correct matrix multiply results (check with large matrices).
- The sequential routine must be at least 6 times faster than the naive implementation in `matmul.c` for  $2000 \times 2000$  matrices on `linprog`.
- The OpenMP routine must achieve a speed up of at least 2 over the improved sequential routine for  $2000 \times 2000$  matrices on `linprog`.
- The MPI routine must achieve a speed up of at least 4 over the improved sequential routine for  $2000 \times 2000$  matrices on `linprog`.

## DEADLINES AND MATERIALS TO BE HANDED IN

- **April 2.** You should demonstrate this program and submit the whole project directory as a tar file. In the directory, you should have a `README` file describing how to compile and run you program and the known bugs in your program.

## GRADING POLICY

- Single-thread program (50)
- OpenMP program (25)
- MPI program (25).

## MISCELLANEOUS

You can get all the optimization techniques for sequential program from the ATLAS paper. You can easily achieve a factor of 6 speed up with various loop optimizations including: memory reordering to improve spatial locality, loop tiling (blocking), loop interchange, loop unrolling; scalar replacement, and instruction scheduling. I got a speed up of 8 with these optimizations in the C source code level.