CS 6230
Term Project
Due 11:59pm, Friday, 12/13/2024

The term project can be done either alone or in teams of two. When grading, some consideration will be given to whether it was a solo effort or a two-person effort, with higher expectations for projects done by two-person teams.

Numerical libraries implement efficient routines for matrix-matrix multiplication (commonly called GEMM – GEneral Matrix Multiplication), where four variants are implemented:

i) **C=AB**, ii) **C=A^{T}B**, iii) **C=AB^T**, and iv) **C=A^{T}B^T**.

For the term project, the **second** (**C=A^{T}B**) and **fourth** (**C=A^{T}B^T**) variants of GEMM are to be optimized by you: 1) Using OpenMP for multi-core CPUs, and 2) Using CUDA for GPUs.

A primary difference between previous optimization exercises for programming assignments and the term project is that of addressing *generality*. For the programming exercises with variants of matrix multiplication, the exercises only involved the "square" case (equal values for all three size parameters for matrix multiplication) and the evaluation involved a fixed power-of-two problem size. The term project highlights the challenge of developing high-performance numerical libraries: high performance is desired across a range of different problem sizes and the specific matrix sizes used by an application program invoking the library routine is not known a priori. The developed codes must execute correctly for any input problem size; thus proper care must be taken with loop unrolling, bounds checking with GPU codes, etc. Further, adaptivity with respect to the problem size parameters in the call can be beneficial. The adaptivity may involve dynamically choosing parameters such as tile sizes, shapes/sizes of thread blocks, or even different code versions, e.g., representing different loop permutations and/or degrees of loop unrolling. If multiple code versions are used, an automatic dynamic selection mechanism must be implemented that chooses one of the versions based on problem size parameters.

The code versions to be optimized are:

1) Transposed-Nontransposed Matrix-Matrix Multiplication (C=A^{T}B):

```
for (i=0;i<Ni;i++)
 for (j=0;j<Nj;j++)
  for (k=0;k<Nk;k++)
// C[i][j] += A[k][i]*B[k][j];
   C[i*Nj+j] += A[k*Ni+i]*B[k*Nj+j];
```

2) Transposed-Transposed Matrix-Matrix Multiplication (C=A^{T}B^T):

```
for (i=0;i<Ni;i++)
 for (k=0;k<Nk;k++)
  for (j=0;j<Nj;j++)
// C[i][j] += A[k][i]*B[j][k];
   C[i*Nj+j] += A[k*Ni+i]*B[j*Nk+k];
```

Template codes will be provided. Below are examples of problem-size variation for test cases. Other sizes may be used in testing the solutions. Report performance on CADE.

Ni	Nj	Nk
16384	8192	256
8192	16384	256
16384	16384	128
8192	8192	512
2048	2048	8192
512	512	131072
256	256	524288

Ni	Nj	Nk
16377	7777	257
7777	16377	257
15677	15677	131
7777	7777	777
1999	1999	33333
511	511	13111
131	131	55557