High Performance Computing: Tools and Applications

Edmond Chow
School of Computational Science and Engineering
Georgia Institute of Technology

Lecture 12

Intel Math Kernel Library (MKL)

- Many functions will create their own team of threads
- You usually *do not* want to call these multithreaded functions from a parallel region
- If calling from a parallel region using all the cores, you usually want to set the number of threads to 1 (library function operates sequentially)
- All functions are thread-safe
- environment variable `MKL_NUM_THREADS` to set *maximum* number of threads used by MKL

LAPACK in MKL

- LAPACK is a FORTRAN multithreaded linear algebra library
- LAPACKE is the C interface to LAPACK
- Function for computing a Cholesky factorization (of a symmetric positive definite matrix)

```
#include <mkl.h>

lapack_int LAPACKE_dpotrf(int matrix_layout,
                          char uplo,
                          lapack_int n,
                          double *a,
                          lapack_int lda);
```

LAPACKE_dpotrf

```
matrix_layout = LAPACK_ROW_MAJOR or LAPACK_COL_MAJOR
uplo          = U or L (use only a triangular portion of a)
n             = number of rows and columns
a             = array of size lda*n containing the matrix
lda           = leading dimension of a
```

- For efficiency, you may want have rows/cols aligned on 64 byte boundaries (use `lda` for this)
- On output, the array `a` is overwritten by the Cholesky factor. Which factor is computed depends on `uplo`.
- Return value `0` means success.
- Positive return value means a negative pivot was encountered.
- Negative return value means a parameter has an illegal value.

C interface to BLAS

Example: compute `C = alpha*op(A)*op(B) + beta*C`

```
#include <mkl.h>

void cblas_dgemm(const CBLAS_LAYOUT Layout,     // CblasRowMajor or CblasColMa
                 const CBLAS_TRANSPOSE transa, // CblasNoTrans or CblasTrans
                 const CBLAS_TRANSPOSE transb,
                 const MKL_INT m,               // C is m by n
                 const MKL_INT n,
                 const MKL_INT k,               // inner dimension
                 const double alpha,
                 const double *a,
                 const MKL_INT lda,
                 const double *b,
                 const MKL_INT ldb,
                 const double beta,
                 double *c,
                 const MKL_INT ldc);
```

Read time stamp counter (Intel compilers)

- High resolution timing could be performed using the `rdtsc` instruction

```
unsigned long int start, stop;
start = __rdtsc();
...
stop = __rdtsc();
```

- To measure how many ticks there are in a second, you could time `sleep(1);`
- It is also possible to access the `rdtsc` instruction on Gnu compilers by inserting assembly instructions

Example

```
#include <stdio.h>
#include <unistd.h> // sleep

void main()
{
    unsigned long int start, stop;

    start = __rdtsc();
    sleep(1);
    stop  = __rdtsc();

    printf("%ld\n", stop-start);
}
```

Killing your jobs

- Log onto the mic coprocessors and kill any of your runaway jobs.
- On joker, your uid on the host and on the coprocessors may be different, so you may have permissions problems. Try:

```
ssh mic0 pkill bd_mic
```

- To check your uid, run `id` on the host and on the coprocessor.