**Performance Analysis of Pthreads, OpenMP, and MPI.**

By Lev Kavs, Sam Moylan, Mitchell Slavens

**Hardware Specifications**

The programs were ran on the same hardware to keep the testing environment constant between the different programs. The machines were constrained to ‘elves’ which contain two 8-Core Xeon E5-2690 processors or a two 10-Core Xeon E5-2690 V2 processors.

**System Specifications**

The OS Beocat is using is the CentOS Linux. The Linux kernel is the 3.10.0 – 957.1.3.el7.x86\_64. The code is compiled using the GCC version 4.8.5 (Red Hat 4.8.5-36). Slurm version is 18.08.6-2

**OpenMP Software Architecture**

The number of threads is passed in as a command line argument. The file is read into memory, each line read from the file is placed into an index of our wiki\_dump array. That array is then iterated over using the ‘#pragma omp parallel for’, this distributes the iterations among the threads. Inside the for loop the call to our ‘algorithm’ is made where it compares the line index passed with the line after. Algorithm is also passed a reference to the wiki\_dump array. Each separate call to ‘algorithm’ processes another line from wiki\_dump. When a substring is found the corresponding index into the ‘longestCommonSubstring’ is allocated and the substring is copied into the place. Once ‘#pragma omp parallel for’ loop is done the time and memory use is collected and printed out.

**Pthread Software Architecture**

The number of threads is passed in as a command line argument and the file is read into memory with the same process as the OpenMP version. A loop iterates through the number of threads. Inside each iteration a algorithmArgs\_t structure is allocated which contains a reference to the longestCommonSubstring array, a reference to the populated wiki\_dump array, and a start and stop index to process of the wiki\_dump array. This struct is passed in as the final argument into the pthread\_create function call. The algorithm is passed as the function to run, ‘algorithm’.