EECS 338 Homework 7

General requirements:

- See posted due date.
- Create a typed report for discussion questions and program output.
- Upload a single, compressed file (e.g. zip) to Canvas that contains all required files (programs + report).
- All work should be your own, as explained in the Academic Integrity policy from the syllabus.
- **Provide SLURM scripts that compile your programs (not makefiles).**

Special assistance: When emailing requests for help using the HPCC, be sure to include the following email address for the admin staff:

hpc-support@case.edu

Extra credit option: In order to reduce competition for on the HPCC, the following extra credit options are offered:

Submitted by...	*Extra credit points*
Monday at 11:59pm	+20
Tuesday at 11:59pm	+10
Wednesday at 11:59pm	*none*

Debugging tip: The "production" version of these problems require SLURM scripts, used in batch mode, that execute the programs multiple times with different parameters. However, it is recommended that you use interactive mode (srun) to test and debug your programs, rather than doing this in batch mode with a SLURM script. If OpenMP works on your personal computer, you may also wish to begin testing there before testing on the HPCC. However, remember that there are known issues with OpenMP on Mac OS.

1. Revise your OpenMP algorithm speedup program from HW #5, problem #3 as follows. You may also use the posted HW #5 solution for the Shubert application.
   - The program should work for an arbitrary number of threads which can be set using the OMP_NUM_THREADS environment variable, as explained previously in class. For example, the following Linux command will cause a program with a directive "#pragma omp parallel for" to automatically use exactly 2 threads, regardless of how many cores are available:
     ```
     export OMP_NUM_THREADS=2
     ```
   - Create a *single* SLURM script that compiles the program on the Case HPCC and runs the program using at least 4 different thread quantities. IMPORTANT: For the highest number of threads used, the run time must be at least one second! You should request the number of cores (cpus-per-task) to be the *maximum* quantity of threads you expect to use. For example, if you choose to use thread quantities 1, 2, 4, and 8, you should request 8 cores. The OMP_NUM_THREADS environment variable can be set to a different value before each execution of the program. For example, the following lines in a SLURM script would compile "program.c" and execute it twice with 1 thread and 2 threads, respectively:
     ```
     gcc program.c -o program -fopenmp -lm
     export OMP_NUM_THREADS=1
     ./program
     export OMP_NUM_THREADS=2
     ./program
     ```
   - **Create a graph of the run times vs. number of cores. Your typed report should include the job output and your graphs (one graph for each problem). Use any graphing software you wish.**

2. Repeat problem #1 using the "critical" clause and a single, shared variable for the overall result, as in HW #5, problem #4. Create a graph of the run times vs. number of cores. Your typed report should include the job output and your graphs (one graph for each problem). Use any graphing software you wish. **Additionally, compare the run times to those you obtained in problem #1 above. Are they lower, higher, or approximately the same?**

3. The program "switch.c" is provided and uses OpenMP to execute the same busy loop on every thread (the work is not divided). In the example below, the program was executed on a system with 2 cores using "/usr/bin/time" and 2 threads (some output has been removed for clarity). As explained in class: user time is computed as the sum of times for individual threads ($t_1+t_2+t_3+t_4+...$), and %CPU is approximately the user time divided by the elapsed (wall clock) time.

```
$ export OMP_NUM_THREADS=2
$ /usr/bin/time -v ./switch
Running...
Running...
        Command being timed: "./switch"
        User time (seconds): 6.49
        Percent of CPU this job got: 198%
        Elapsed (wall clock) time (h:mm:ss or m:ss): 0:03.26
        Involuntary context switches: 169
```

PART 1: Create a *single* SLURM script that does the following on the Case HPCC. **Your typed report should include the job output.** You may include all of the output, or you may omit extra lines, as shown in the above example.

- Requests 4 cores and compiles the program.
- Runs the program using thread quantities 2, 3, 4, 5, and 6.

PART 2: Provide the following in your **typed report.**

- Make a table that shows the following from PART 1 for the different thread quantities:
  - ✓ number of threads
  - ✓ user time
  - ✓ %CPU
  - ✓ elapsed time
  - ✓ involuntary context switches
- Explain the changes in %CPU and involuntary context switches with regard to the number of cores and number of threads.