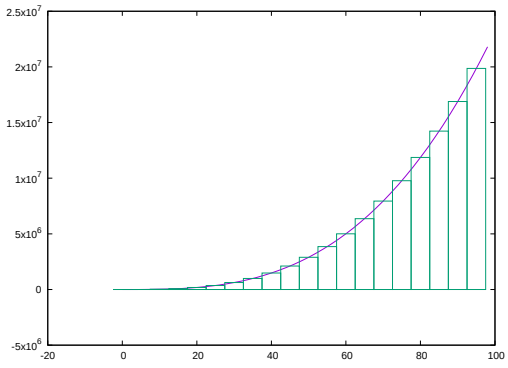
ECE2036 **Fall Semester, 2016**

Parallel Numerical Integration

Assigned: April 11, 2016 *Due: April 25, 2016 11:59pm*

Overview. This assignment will use 10 threads to compute in parallel the following integral:

$$\int_0^{+100} 23.864x^3 + 5.7x^2 - 173.1x + 10.523dx$$



Program Requirements.

1. The specified integral is to be computed using a numerical technique called the *Riemann Sum*. The Riemann sum estimates the value of an integral by computing the area of a series of rectangles at equidistant points along the x–axis. The Riemann sum is characterized by the Δx value, which specifies the horizontal size of each rectangle along the x–axis. The height of each rectangle is the average of $f(x)$ and $f(x + \Delta x)$. The figure below shows the specified integral with Δx value of 5.0. Since we are integrating between 0 and 100.0, this value of Δx results in the summation of 20 rectangles to get the final value of the integral. In general, it is expected that smaller Δx value result in more accurate results at the expense of additional computation.

2. The program must compute the Riemann sum using 10 threads.

3. The value of the integral must be calculated 7 times, using varying Δx values, starting from 1.0 and ending with 0.000001, dividing by 10.0 for each iteration. This means the number of rectangles summed varies from 100 to 100,000,000.

4. The `Makefile` provided actually compiles the program twice and makes two different executable binaries. The first one uses a 32–bit `float` value for all floating point values. The second one uses a 64–bit `double` value for the floating point values. The executable binaries are called `integration-float` and `integration-double` respectively.

5. Since the integral we are computing is in fact analytically integrable, we can compute the "correct" value as:

$$5.966x^4 + 1.9x^3 - 86.55x^2 + 10.523x + C|_0^{+100} = 5.9763555230000e + 08;$$

6. For each of the seven different Δx values, print out the Δx, the computed integral, and the error quantity (the absolute value of the computed value minus the correct value). Use the `printf` function for this, with the format string "%3.6f" for the Δx value, and "%15.13e" for both the computed area and the error value.

Obtaining the skeleton files. Once logged into deepthought19, you can get all of the required materials for this assignment by running the command:

```
rsync -avu /nethome/ECE2036/NumericalIntegration .
```

DON'T FORGET THE PERIOD AT THE END OF THE RSYNC COMMAND. The final argument above (the period) tells `rsync` where to write the files. The period indicates "the current working directory".

This will create a subdirectory called `NumericalIntegration` with the files you need.

Copy the skeleton `integration-skeleton.cc` as follows:

```
cd NumericalIntegration
cp integration-skeleton.cc integration.cc
```

Subroutines provided in `gthread.h`

1. `CreateThread();`
   See the discussion of `CreateThread` in the lecture notes handout on threads.

2. `EndThread();`
   See the discussion of `EndThread` in the lecture notes handout on threads.

3. `WaitAllThreads();`
   See the discussion of `WaitAllThreads` in the lecture notes handout on threads.

4. `LockMutex(gthread_mutex_t m);`
   Claims the lock on the specified mutex. See the discussion of `LockMutex` in the lecture notes handout on mutual exclusion.

5. `UnlockMutex(gthread_mutex_t m);`
   Releases the lock on the specified mutex. See the discussion of `UnlockMutex` in the lecture notes handout on mutual exclusion.

   IMPORTANT NOTE: Variables of type `gthread_mutex_t` should be initialized to NULL. If the mutex is a global variable this happens automatically.

Turning in your program As before, use the turnin-ece2036a or turnin-ece2036b script.