Introduction to Parallel Computer Architecture
Numerical Integration

Prof. Naga Kandasamy
ECE Department, Drexel University

April 9, 2019

The problem, worth 10 points, is due April 21, 2019, by 11:59 pm via BBLearn. You may work on this problem in a team of up to two people. One submission per group will suffice. Please submit original work. Solutions copied from other students or from online sources will result in a grade of zero for the assignment. You may discuss high-level approaches to solve the problem with Shihao Song or me, but we will not help you with the code itself.

Given a function $f(x)$ and end points a and b, where $a < b$, we wish to estimate the area under this curve; that is, we wish to determine $\int_a^b f(x)\,\mathrm{d}x$.

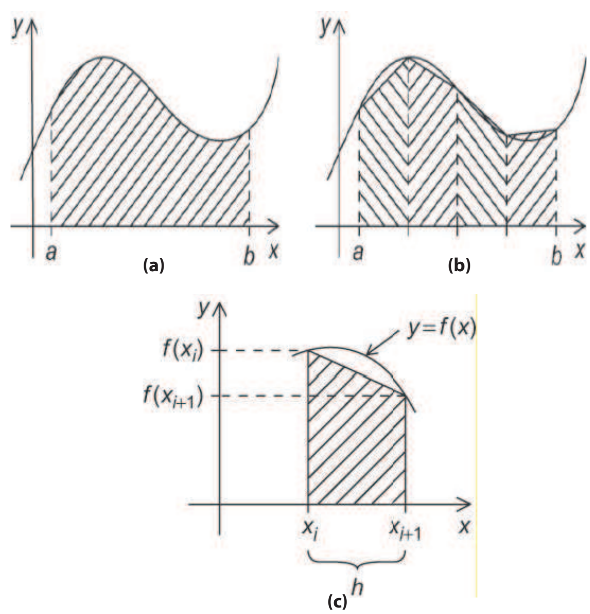

Figure 1: Illustration of the trapezoidal rule: (a) area to be estimated; (b) approximate area using trapezoids; and (c) area under one trapezoid.

The area between the graph of $f(x)$, the vertical lines $x = a$ and $x = b$, and the x-axis can be estimated as shown in Fig. 1 (b) by dividing the interval $[a, b]$ into n subintervals and approximating the area over each subinterval by the area of a trapezoid. Fig. 1(c) shows one such trapezoid where the base of the trapezoid is the subinterval, its vertical sides are the vertical lines through the endpoints of the subinterval, and the fourth side is the secant line joining the points where the vertical lines cross the graph. If the endpoints of the subinterval are x_i and x_{i+1}, then the length of the subinterval is $h = x_{i+1} - x_i$, and if the lengths of the two vertical segments are $f(x_i)$ and $f(x_{i+1})$, then the area of a single trapezoid is $\frac{h}{2}[f(x_i) + f(x_{i+1})]$. If each subinterval has the same length then $h = (b - a)/n$. Also, if we call the leftmost endpoint x_0 and the rightmost endpoint x_n, we have

$$x_0 = a, x_1 = a + h, x_2 = a + 2h, \ldots, x_{n-1} = a + (n-1)h, x_n = b,$$

and our approximation of the total area under the curve will be

$$\int_a^b f(x)\,\mathrm{d}x = h[f(x_0)/2 + f(x_1) + f(x_2) + \cdots + f(x_{n-1}) + f(x_n)/2].$$

Thus, the code for a serial algorithm looks like the following.

```
double trapz (float a, float b, int n)
{
    float h = (b - a)/n;
    double sum = (f (a) + f (b))/2.0; /* f is function of interest. */
    float x;

    for (int i = 1; i <= n - 1; i++){
        x = a + i * h;
        sum += f (x);
    }

    sum = h * sum;
    return sum;
}
```

The program given to you accepts four command-line arguments: the lower and upper limits of integration, a and b, respectively, the number of trapezoids n, and the number of threads. The function $f(x)$ is defined within the file `trap.c` as

$$f(x) = \sqrt{\frac{1 + x^2}{1 + x^4}}.$$

Edit the `compute_using_pthreads()` function within the file `trap.c` to complete the required functionality on a multi-core CPU. Upload all of the source files needed to build your executable as a single zip file to the BBLearn site. Also, include a brief report describing: (1) the design of your multi-threaded implementation, providing code or pseudocode to clarify the discussion; and (2) the speedup achieved over the serial version for 2, 4, 8, and 16 threads. The report can include the names of the team members on the cover page.