P. Koumoutsakos, P. Hadjidoukas
ETH Zentrum, HG D1.1/G3
CH-8092 Zürich

Set 3 - OpenMP Tasks

Issued: March 12, 2018
Hand in: March 19, 2018

Question 1: Parallel Monte Carlo Numerical Integration

Code in `montecarlo/mc_serial.cpp` computes the integral of a function using the Monte Carlo method.

Implement in `mc_parallel.cpp` a parallel version of the code using **OpenMP tasks**.

Examine the scaling of your code.

Question 2: Parallel Adaptive Integration

Adaptive integration is the numerical integration method in which the integration interval is recursively subdivided into smaller subintervals based on the accuracy of the underlying basic integration method (like trapezoidal rule) for that subinterval. The method allocates more function evaluation in the regions where it behaves "badly" and saves evaluations where the function is "nice" and smooth. The error of the trapezoidal rule is estimated by comparing the estimate with a coarser estimate.

The 1D adaptive algorithm for computing $I = \int_a^b f(x)\,dx$ with desired accuracy τ looks as follows:

function ADAPTIVE(f, a, b, n, τ)
  $Q \leftarrow$ TRAPEZ(f, a, b, n)
  $Q^* \leftarrow$ TRAPEZ(f, a, b, $n/2$)
  $err \leftarrow |Q - Q^*|$
  if $err > \tau$ **then**
    $mid \leftarrow \dfrac{a+b}{2}$
    $l \leftarrow$ ADAPTIVE(f, a, mid, n, $\tau/2$)
    $r \leftarrow$ ADAPTIVE(f, mid, b, n, $\tau/2$)
    return $l + r$
  else
    return Q

Here n is the number of steps in trapezoidal rule.

a) Write an adaptive integration algorithm for a 2D integral.

b) In the file `adaptive/adaptive_integrator_serial.cpp` you can find a simple adaptive integration code for two dimensions.

Implement in `adaptive_integrator_parallel.cpp` a parallel version of the code using **OpenMP tasks**.

Examine the scaling up to 24 cores at least for 10 and 100 segment samples and a target error of 10^{-7}. Comment on the results.