Adaptive Task Partitioning in *ParInt*
Omofolakunmi Olagbemi (with Dr. Elise de Doncker)
College of Engineering and Applied Sciences, Western Michigan University, Kalamazoo

**Introduction**
*ParInt* is a software for parallel multivariate integration, and was developed by a group of Computer Science (CS) Faculty members and PhD students led by Dr. Elise de Doncker. The project was funded by the National Science Foundation (NSF) through a series of grants.

**Problem definition**

*ParInt* (represented by the black box in the diagram above) includes an adaptive algorithm in integrating functions specified by the user, and repeatedly evaluates the results till the user’s requested accuracy is attained (that is, tolerated error is not exceeded) or till it has reached or exceeded a pre-determined limit on number of evaluations (computations) to be performed.

*Input*  
Function $f$  
Region $D$  
Tolerated error $tol$

*Output*  
Integral approximation $Q$  
Error estimate $E$

Compute:  
$Q \approx \int_D f(x) \, dx$, $E \approx |Q - I|$

such that  
$|Q - I| \leq E \leq tol$

This diagram shows a plot of the integrand function below for $\varepsilon = 0.1$

\[
\int_0^1 \int_0^1 dy \frac{2 \in y}{(x+y-1)^2 + \in^2}
\]

**Adaptive integration meta-algorithm**

| Evaluate initial region and update results  
| Initialize priority queue with initial region  
| while (evaluation limit not reached and estimated error too large)  
| Retrieve region from priority queue  
| Split region  
| Evaluate new sub-region and update results  
| Insert new regions into priority queue |

We consider the function 1 below:

\[
f(x, y, z) = x^{-0.2} y^{-0.2} z^{-0.2} (x + y + z)^{-0.2} \quad (1)
\]

**Results**

The chart above shows the execution times when computing the integrand for function (1) as well as its 4- and 5-dimensional versions. Evaluation limits of 10 million, 20 million and 40 million were used. These were executed on the clusters in the High Performance Computational Sciences (HPCS) Laboratory in the Department of Computer Science (CEAS). From the results above, using an increasingly higher number of processes resulted in progressively lower execution times, until excessive overheads prevent further decrease of execution time.

**Applications**

1. Statistical analysis of finite elements (for example, automotive simulations),
2. Computational finance (financial derivatives, for example, collateral mortgage obligations, mortgage backed security problems),
3. High energy physics (modeling of particle interactions),
4. Computational chemistry (for representing the electronic structure of an atom or molecule),
5. Computational geometry (tessellations),
6. Biometrics (for example, for analysis of taste-testing trials)

**Future work**

1. Extensions for a hybrid environment (with distributed computing, multi-core, many-core on Graphical Processing Units - GPUs).
2. Extensions with respect to numerical methods.
3. Extensions with respect to parallel strategies and load balancing using hybrid architectures.

**References**

1. Shujun Li; Online Support for Multivariate Integration; PhD Thesis; Western Michigan University; 2005.