Project Name: MultiPhysics-Sim
Project Title: Scalable Multiphysics Simulations with OpenFoam & Nek5000
Institution: ANL
Division: CPS
Associated Funding: Applied Math Research Program, BES, LDRD
Other Systems: ALCF (Mira/Theta), NERSC, SDSC (Comet), TACC (Stampede 2).

Science: Our mission is to develop and deploy scalable algorithms for a variety of multiphysics applications using open-source codes. Our primary vehicle for algorithm development are open-source codes, OpenFoam and Nek5000. OpenFoam is a finite-volume multiphysics solver/library written in C++ and is a spectral element fluid/thermal simulation code written in Fortran. Both these codes have shown good strong and weak-scaling on multi-dimensional CFD problems with millions of grid-points and are widely used by research groups the world over, and hence the focus of our study.

Project description: Multiphysics/multidisciplinary simulations refers to the numerical study of problems involving multiple simultaneous physical phenomena. Such problems span a wide range of problems in engineering applications such as reacting flows in gas turbines, IC engines, combustors which involve a strong interaction between flow, heat-transfer, chemistry (combustion kinetics) and structures (thermal stress, flow-induced vibrations etc). Similarly, semiconductor processing reactors involve the use of electromagnetic fields for the deposition of materials on substrates. These reactors are more complex, involving an interaction between flow, chemistry (both surface and gas-phase) and electromagnetics. Multiphysics problems are also multi-scale, in that the various processes occur at time-scales that are widely varying in magnitude. For instance, the typical flow-through time in a combustor would be on the order of few hundred milliseconds, whereas the chemistry occurs at tens of micro-seconds. Developing scalable, predictive simulation capabilities for a large-class of multi-physics simulations is of interest to various DoE missions and also industry. Development and demonstration of robust, scalable computations on large-scale clusters, reduced wall-time to solution and thus greatly accelerates the process of discovery and product development.

Software requirements: Both OpenFoam & Nek5000 use MPI. Various intel/gcc compilers and MPI/OpenMPI flavors available on Bebop meet our compilation needs. VisiT and Paraview are the open-source visualization software required for Nek5000 and OpenFoam which are available on Bebop.

Experiments planned & size of calculations:

In this project, we anticipate running about running about 10 OpenFoam simulations each on 512 KNL cores (8 nodes) and 5 Nek5000 simulations on 64 Broadwell nodes (2048 cores) each quarter. A typical time-step takes about 17 secs on 512 cores of the KNL. We need about 10,000 time-steps to run for each case to completion, requiring about 48 hours/case.

The Nek5000 simulations take about 48 seconds per time-step on 64 broadwell nodes (2048 cores). We need about 2000 time-steps per case, requiring about 24 hours/case.

Total time request:
OpenFoam (KNL) = 10 x 512 * 48 = 250,000 core-hours (approx)
Nek5000 (Broadwell) = 5 x 2048 * 24 = 250,000 core-hours (approx)

Total request = 500,000 core-hours/quarter

There will be 3-4 ANL members and two industry partners in this project.

Industry Partnership:
We are closely collaborating with ACK industries on this project. Two project engineers from ACK industries will be working closely with the ANL team. Their role in the project would be to run simulations on LCRC clusters and also running smaller problems with the ANL problem set-up on their in-house clusters.

URL: http://cps.anl.gov/~scale_algo

Large Allocation efficiency:
Scaling/Performance: We have conducted a detailed scaling study on a typical problem size of interest to this project on the broadwell & KNL nodes. Our typical problem of interest is about 10-100 million grid points.
OpenFoam

Title: Scaling on KNL for an injector problem with 12 million grid-point - OpenFoam.

Cores	Time per time-step (sec)	Scaling efficiency
128	76	100%
256	38	100%
512	20	95%
1024	15	63%

It is seen that we have good strong scaling (95 % parallel efficiency) up to 512 cores on the KNL cores and hence our typical problem will be run up to 512 cores.
Note to PIs. A scaling study is done with only a few time-steps (or iterations) and should typically be run so it takes about 10 minutes on a single node. If your problem needs more than 10 minutes per time-step on a single node then run it for one time-step. Run the same problem on 2 nodes and compute parallel scaling efficiency with the formula*
*P_eff = (time for computations on 1 node*100/(N*time for same computations on N nodes). We recommend that you continue the scaling study by doubling the number of cores till you reach 70% parallel efficiency. In the above example, we started the scaling study on 2 KNL nodes, at 4 nodes (256 cores), we see that doubling the number of cores reduces the compute time by half – so we have 100% or ideal scaling. When we double the number of cores again the time is*

reduced from 38 sec to 20 sec but beyond 512 cores, the parallel efficiency is 63%. So this problem should be run on 512 cores for optimal use of the cores.

Nek5000:
We demonstrate strong scaling for another targeted problem with about 10M grid-points on Broadwell nodes. In this example we ran the same problem for 500 steps from 4 nodes (128 cores) up to 128 nodes (4096 cores).

Title: Scaling on Broadwell nodes for a problem with 10 million grid-point - Nek5000

Nodes (cores)	Time for 500 time-steps (secs)	Scaling efficiency
128 (4096)	32	63%
64 (2048)	57	71%
32 (1024)	107	75%
16 (512)	195	83%
8 (256)	368.	88%
4 (128)	647.8	100%

It seen that the problem shows good strong scaling (approx 71 %) up to 64 nodes. At 128 nodes, the parallel efficiency drops to about 63% - so all our production runs will be at 64 nodes.
Note to PIs. In the above problem, we have used the same procedure as above starting with 4 BDW nodes and determine that the problem is best run at 64 nodes. To estimate your core-hour requirements for runs on 64 nodes, assuming your problem converges in ~50,000 time-steps, you would need ~57*(50,000/500)*64*32/3600 ~3240 core-hours.*
Storage requirements: While we will be generating large data sets, the project members will review solutions file on a monthly basis to retain only important case-specific files. 1 TB storage will be sufficient.