LA-UR-18-26679

SNAP

Structured mesh, linear radiation pseudo-transport proxy application

Benchmark Problem Description and Specifications

SNAP solves a structured mesh, linear radiation pseudo-transport problem. This specific problem is of a fixed, 3-D spatial domain with a fixed discretization in the x-dimension. The y- and z-dimension discretization are weakly scaled with the number of nodes to create the benchmark at varying resource allocations. The number of energy groups and angular directions for mesh sweeps are fixed at 54 and 48, respectively. The material map and source map are fixed. The problem will run for two time steps.

The job has been designed to take approximately 40% or less of the memory available per node. **A hard limit of no more than 50% of the memory available per node should be used.** Further, if a whole system is to be used, then SNAP should be limited to no more than 50% of the entire memory available and preferably below the 40% design here. A suitable floor to this limitation is 25%. To assist in meeting this limitation output was added to SNAP to report the number of words allocated per rank.

Code Access and Compilation Details

This is a special version of SNAP, pre-release to public at GitHub. It features a few key differences that those familiar with SNAP should be aware of.

1. The code has a new input called “cor_swp.” All benchmark runs were performed with cor_swp=1. This option uses fully asynchronous communications during the compute intensive “transport sweep” kernel, that exposes greater concurrency and can improve performance.
2. The number of allocated double precision words per MPI rank has been summed and provided at the bottom of the output file. This value can be used to estimate memory utilization per computational resource.

SNAP was compiled with Intel 2017.0.4 (Fortran) with OpenMP extensions and Cray MPICH 7.7.0. Used -xHost to get AVX2 instructions for a Haswell build using the provided Makefile. SNAP should present no issues for Fortran compilers with OpenMP extensions.

Execution Details

Results below present 1-node, 64-node, and 4,096-node problems that are related through a weak scaling of the spatial domain. Each problem was run on the Haswell partition of Trinity. Each of these types of runs was done with MPI only, 2 threads/rank, and 4 threads/rank. Affinity was set at the core level for threads. That is, as threads were increased, MPI ranks were not fixed; an increase in the number of threads per rank was met with a proportional decrease in the number of ranks. Each run was performed multiple times and the results shown below are the median of those runs. Noise in execution time was generally very small except for a couple extreme outlier cases that were assumed to be the result of some system perturbation/slow node issue.

Specific SLURM command I used for rank/thread mapping and affinity:

```
srun -n # ranks -c #threads*2 –cpu_bind=cores …
```

Note that this -c usage permits the threads to migrate over the Haswell hardware threads as the runtime system sees fit. I investigated other options to more carefully bind threads but did not observe significant difference.

Results

Node Type	Nodes	Ranks	Threads/Rank	Iterations	Solve (s)	Grind (ns)
Haswell	1	32	1	2226	102.06	7.29e-1
Haswell	1	16	2	2226	98.27	7.02e-1
Haswell	1	8	4	2226	99.99	7.14e-1
Haswell	64	2048	1	2423	191.12	1.96e-2
Haswell	64	1024	2	2423	129.75	1.33e-2
Haswell	64	512	4	2423	130.45	1.34e-2
Haswell	4096	131072	1	2689	282.95	4.08e-4
Haswell	4096	65536	2	2689	183.36	2.65e-4
Haswell	4096	32768	4	2689	203.83	2.94e-4

Correctness

The number of iterations increases as the problem is weakly scaled, probably because of the fabricated numerical formulas used in SNAP. To ensure correctness it is strongly recommend to confirm that the number of iterations matches with some very small leeway (+/-10 iterations). Because SNAP does not solve a real problem, further results do not reveal much additional insight into the "correctness" of the code. However, large deviation in the number of iterations likely indicates some aspect of the code is not working correctly and should be further investigated. The line in the SNAP output can be found by searching for the unique string, "Total inners", which reports the total number of full transport mesh sweeps performed (the most important kernel).

Figure of Merit

The reported "Solve" time is the Figure of Merit, with correctness checked through the total number of iterations and staying within the bounds of the memory per node constraint. This provides the most relatable sense of how the code is performing.

The FOM and constraints can be found in the SNAP output:

```
grep "Solve"
grep "Total inners"  [must be 2689 +/- 10]
grep "Allocated words"  [must be less than 50%]
```

Reporting

For the electronic submission, include all the source and makefiles used to build on the target platform and input files and runscripts. Include all standard output files.