Perlmutter performance optimization

S. Scott (CFS)

May 15, 2023

This file: Perlmutter_performance_optimization.docx

The purpose of this exercise is to determine the optimum run-time switches for the ASCOT5 orbit-following code on Perlmutter and to compare the performance on Perlmutter versus KNL.

#	Group	Nmrk	.sh file	runID	--nodes	--ntasks	--cpus-per-task	OMP_NUM_THREADS	Wall time
1	group_go_2456	250,000	perlmutter_2456d	(none)*	4	2	128	64	(none)
2	group_go_2456	250,000	perlmutter_2456e	8825501	4	4	128	64	18:05
3	group_go_2456	250,000	perlmutter_2456f	8826613	4	4	128	128	14:34
4	group_go_2456	250,000	perlmutter_2456g	8827184	4	4	128	256	15:46
5	group_go_2456	250,000	perlmutter_2456h	8827230	4	4	128	512	16:09
6	group_go_2456	250,000	knl_2456	3068164	4			134	40:33

Table 1. Results of run-time-switch optimization on Perlmutter for ASCOT5.

*Note: simulation #1 was disallowed: Nodes=4 and ntasks=2 is forbidden.

All of the Perlmutter simulations use the executable:

/global/cfs/cdirs/m3195/ascot/ascot5-perlmutter-jw/ascot5/ascot5_main_intel

which has size = 1,787,552 and time-stamp = 9/1/2022

Comparing the fastest simulation on Perlmutter to KNL, we get a speed-up of about a factor of 2.8 which I think is about the expected speed-up (see below). I could do a similar optimization on knl, but I think the chosen value of OMP_NUM_THREADS (34) for the knl simulation was chosen from a previous optimization on knl. Since we are migrating to Perlmutter anyway, and since we're getting about the right speed-up, there seems little reason to pursue this much further.

xpected performance vs knl

	Cori Haswell	Cori KNL	Perlmutter CPU
Physical cores	32	64	128
Logical CPUs per physical core	2	4	2
logical CPUs per node	64	272	256
clock speed (GHz)		1.4	2.45
NUMA domains	2	1	8
-c value for srun	floor(tpn)*2	floor(68/tpn)*4	floor(128/tpn)*2

Assuming that multi-threading does not provide much performance enhancement, the ratio of performance of Perlmutter to knl would be: R = (128 * 2.45) / (68 * 1.4) = 3.29.