Lightweight threading and vectorization with OpenMP in ACME
Azamat Mametjanov, Robert Jacob, Mark Taylor

Objective

Next-generation DOE machines Cori II (October 2016) and Aurora (2018) are expected to provide 3.5x more cores per node with KNL and KNH Xeon Phi many-core chips. Applications are required to increase their fine-grained threading and vectorization potential to fully occupy ~60 cores each with 8-double-wide vector units. While the new on-package high-bandwidth memory is expected to provide 5x speedup to existing executables, significant source code refactoring and fine-grained parallelization is needed to achieve better efficiency. Pure MPI parallelization can lead to network congestion degradation and out-of-memory errors at high number of tasks per node. Our goal is to leverage shared-memory parallelism of OpenMP to efficiently utilize available hardware threads and vector units.

Primary methods toward this goal:
1. Parallelize sequential loops
2. Collapse nested loops
3. Vectorize innermost loops

Complicating factors are threading and SIMD overheads, bit-for-bit correctness and portability of speedups and correctness across supported compilers – Intel, IBM, PGI, GNU.

Approach

Profile to pick compute-bound regions
- CrayPAT, HPCToolkit, Intel VTune/Advisor/Inspector, ...
- No "hot" regions: many "warm"

OpenMP updates (in diff format)
call t_start(f"omp_par_do_region")
!$omp parallel do num_threads(16), default(shared), private(ie) ! parallelize
do ie = nets, note
    call omp_set_num_threads(vthreads)
    !$omp parallel do private(q,k) collapse(2)
do q = 1, qsize
    do k=1, nlev
    !$omp simd
        gradQ(:,::,1) = Vstar(:,::,1,k) * elem(ie)%state%Qdp(:,::,k,q,ie)
call t_stop(f"omp_par_do_region")

Benchmark: based on env_mach_pes.xml, improve timing_stats
- Also, inspect compiler-generated listings and optimization reports
Test: (PET, PEM, PMT, ERP tests) on (F-, I-, C-, G-, B-cases) without cprnc diffs
Port: re-bench and re-test on other compilers

Impact

Enabled nested threading in ACME
- Nested threading shows speedups: e.g. 1.71x speedup in atm/ind coupled F-cases.
- Beyond scaling limits, nested threads occupy "unused" cores: e.g. ne120 on 8K Mira partition

Improving OpenMP run-time support
- BOLT (BOLT is OpenMP over Lightweight Threads) provides 1.6x speedup to nested threading
- Intel is using transport_se mini-app to improve libraries and compiler support

Early access to KNL nodes
- 64 cores and 256 hardware threads with 16 GB high-bandwidth memory
- Nested threads are able to use all 256 hardware threads while fitting into 16 GB HBM

ACME Accelerated Climate Modeling for Energy
Azamat Mametjanov
For additional information, contact
Argonne National Laboratory
(630) 252-1074
azamat@anl.gov
climatemodeling.science.energy.gov/acme