UMT for Crossroads

2018-07-04

Contents

1 LLNL MARKINGS

```
UMT2013
LLNL-CODE-638452
Title: UMT, Version: 2.0
```

2 NOTICE 1

This work was produced at the Lawrence Livermore National Laboratory (LLNL) under contract no. DE-AC-52-07NA27344 (Contract 44) between the U.S. Department of Energy (DOE) and Lawrence Livermore National Security, LLC (LLNS) for the operation of LLNL. The rights of the Federal Government are reserved under Contract 44.

3 DISCLAIMER

This work was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor Lawrence Livermore National Security, LLC nor any of their employees, makes any warranty, express or implied, or assumes any liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately-owned rights. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

4 NOTIFICATION OF COMMERCIAL USE

Commercialization of this product is prohibited without notifying the Department of Energy (DOE) or Lawrence Livermore National Security.

5 Introduction

This documentation is an addition to the initial `README` file that came with UMT for the ATS-3 procurement. That file's contents have been included and updated within this document.

UMT2015 is an LLNL ASC proxy application (mini-app) that performs three-dimensional, non-linear, radiation transport calculations using deterministic methods. The method of solution is as follows. UMT performs the solution of time-dependent, energy-dependent, discrete ordinates, and nonlinear radiation problems on three-dimensional unstructured spatial grids on large distributed-memory multi-node parallel computer systems with multiple cores per node. To achieve extreme scalability, the application exploits both spatial decomposition using message passing between nodes and a threading algorithm in angle within the node.

The project addresses the need to obtain accurate transport solutions to three-dimensional radiation transfer problems (i.e. the transfer of thermal photons). The solution calls for the use of an unstructured grid. This class of problems is characterized by tens of thousands of unknowns per zone and upwards of millions of zones, thus requiring large, scalable, parallel computing platforms. The package uses a combination of message passing and threading, utilizes the large distributed memory of the platform and provides unprecedented (weak) scaling to very large core counts.

The application will execute on all modern High Performance Computing (HPC) systems. At LLNL we have been using IBM BG/Q systems and Intel Xeon (Sandybridge) clusters to develop and test the software.

The number of MPI ranks that can be launched on a compute node is limited by the size of the memory available on the node.

The package is designed to take advantage of MPI message passing as well as threading using OpenMP. Consequently systems that have efficient MPI and OpenMP software are better suited for this application, especially at scale.

It is best if the compute nodes have enough DRAM to provide about 1GB or 2GB per core. For systems with 1GB of memory per core, fewer MPI ranks will fit in the node than there are cores, for moderate size problems. Memory footprint will be discussed elsewhere in this file.

The time required to solve problems is highly dependent on the details of the problem, but the software is designed to reduce time to solution with the use of a spatially decomposed and threaded algorithm.

The code is designed to use MPI and OpenMP. SIMD and/or other vectorization may be turned on and is encouraged.

The source code is very similar to that distributed as part of the CORAL procurement. Three changes were made in Feb 2014 to correct thread safety issues.

6 Building UMT

The overall instructions to obtain and build HPCG are enumerated below.

1. Modify `make.defs` to reflect the platform's compilers, compiler options, libraries, MPI wrappers, etc.; `make.defs.bgq` is an example for LLNL Blue Gene/Q systems and `make.defs.Intel_x86_64` is an example for LLNL Intel x86_64 Infiniband clusters.

   - The following options are typically set within `make.defs`:
     - `F90FLAGS_OPT = -g -O3 -no-prec-div -fPIC -openmp`
     - `CXXFLAGS_OPT = -g -O3 -w -no-prec-div -fPIC -openmp`
     - `CFLAGS_OPT = -g -O3 -w -no-prec-div -fPIC -openmp`

2. `gmake veryclean`

3. `gmake`
    - The output from `gmake` is saved in `bld.msg`.
    - The following libraries or their `.a` counterparts should exist at this point:
        - `./CMG_CLEAN/src/libcmgp.so`
        - `./Teton/libTetonUtils.so`
        - `./Teton/libInfrastructure.so`
        - `./cmg2Kull/sources/libc2k.so`
4. `cd ./Teton`
5. `gmake SuOlsonTest`
    - The output from `gmake` is saved in `bld.msg`.
    - The following executable should exist at this point:
        - `SuOlsonTest`

7 Running UMT

ATS-3 test problems are in the `ATS3` sub-directory. The tests run the `SuOlsonTest` executable in the `Teton` sub-directory.

7.1 Scaling Input Decks

It is straightforward to weak scale one of these tests to a different number of MPI processes. First, make a copy of a grid file and edit it.

Roughly equal numbers of processes in each direction are preferred. The number of energy groups and the number of polar and azimuthal angles should not be changed for these runs.

The `sms` card specifies the domain decomposition, i.e., `sms(10,12,14)` means 10 domains in x by 12 domains in y by 14 domains in z.

The `blk` line sets the number of MPI domains in each direction. For 36 domains, the entry is `0:35`. For 4 domains it is `0:3`.

The `tag` lines specify boundary faces. For 36 domains, the entries are 0 or 36 (the upper limit is one greater than for the `blk` line). For the earlier `sms` example, the values for x are 0 and 9, 0 and 11 for y, and 0 and 13 for z.

The `numzones` card sets the number of zones per domain in each direction. The sample problems use 3x3x4 zones per domain for pure MPI runs.

It is possible to strong scale a test problem by setting `OMP_NUM_THREADS` to a number greater than 1 and not changing anything else. There may not be enough work (work is proportional to the number of zones) per domain for efficient operation with only 36 zones. `grid16MPI_3x6x8.cmg` increases the number of zones per domain to 144, enough that 4 threads per domain will lead to the same work per thread as a 3x3x4 pure MPI run.

7.2 Memory Considerations & Use

The UMT test is intended to represent NNSA simulations that use large amounts of memory. Do not be fooled by the seemingly small number of zones. Each zone has 200 groups times 90 angles per octant times 8 octants. The variables are specified at 8 points per zone. The pure MPI tests have 36 zones per domain. The information communicated from other zones may require more memory than the zones "owned" by the process.

A domain uses slightly under 2 GB of memory for the 3x3x4 cases (i.e., benchmarking on CTS-1 reveals this value to actually be 1.7GB) and 2.75 GB for the 4x4x4 cases. The plan for the ATS-3 acceptance testing is that UMT will run on roughly one third of the ATS-1 Trinity Phase 1 (Haswell) system and utilize 0.4PB for the baseline case.

Running with fewer domains and many threads per domain is fine. Reducing the number of zones per domain may lead to lower FOM values due to heavy message passing demands (the surface-to-volume ratio becomes very poor with less than 36 zones).

The memory used to store the radiation field is proportional to the number of zones times the number of groups times the number of angles. There is a value at each of the 8 corners of the hexahedral zones and 2 copies of the radiation field are required. The memory per domain required to store the radiation field for the `run3x3x4_g200_p9a10_D64T1.bash` test problem is:

```
(3x3x4 zones) x (200 groups) x (9x10x8 directions) x (8 corners) x (2
copies) x (8 bytes/value) = 664 MB
```

UMT also needs the radiation intensity from adjacent domains at each corner of the exterior faces of the domain (MPI messages supply the values). A 3x3x4 zone domain has 264 corners on the boundary. Each corner requires 264x200x720 values or 304 MB. This accounts for a total of 968 MB per process. Actual memory usage is a bit under 2 GB per domain.

The remaining memory usage is presumably due to the executable, the operating system, temporary buffers, and MPI buffers.

At any rate, use 2 GB per domain for 3x3x4 zone runs to compute the number of domains that can fit in a given amount of memory.

7.3 Weak Scaling

Weak scaling is performed by holding the number of zones, groups, angles, and angles per domain fixed and increasing the number of domains. If thread synchronization overhead is low and MPI bandwidth is high, UMT should scale well to very large runs.

7.4 OpenMP Parallelism

OpenMP parallelism is deployed in the Fortran 90 function `snflwxyz` in file `Teton/transport/Teton/snac/snflwxyz.F90`. The available parallelism equals the product `Polar*Azim` (i.e., 90 in Example 1).

7.5 Verification & Validation

run3x3x4_g200_p9a10_D64T1-IntelSB.out contains the output (STDOUT and STDERR) from a run of run3x3x4_g200_p9a10_D64T1-IntelSB.bash on 4 nodes of an Intel Sandy Bridge with two sockets and 8 cores per socket. run3x3x4_g200_p9a10_D64T1-IntelSB.dump is the dump.out file produced by this run.

7.6 Examples

Example 3 below is sized for a near full system run on the Edison system at NERSC. It can be used as a starting point for estimating the performance of UMT on the ATS-3 system. This run should require a total of roughly 93 TB of memory.

We have included several run scripts to launch SuOlsonTest:

- run3x3x4_g200_p9a10_D1T1.bash
- run3x3x4_g200_p9a10_D1T16.bash
- run3x3x4_g200_p9a10_D1T4.bash
- run3x3x4_g200_p9a10_D46656T1.bash
- run3x3x4_g200_p9a10_D64T1.bash
- run3x6x8_g200_p9a10_D16T4.bash

The file names encode the number of zones per domain, the number of energy groups, the number of polar and azimuthal angles in an octant, the number of domains, and the number of threads per domain. All test problems are three-dimensional.

run3x3x4_g200_p9a10_D64T1.bash and run3x6x8_g200_p9a10_D16T4.bash have the same total number of zones and are intended for the same total number of cores (64).

Here are some examples of running UMT on a cluster with two Intel Sandy Bridge 8-core processors per node are included. This cluster uses SLURM job scheduling (e.g., srun).

7.6.1 Initial Crossroads Example 1 (1 Node, 1 MPI Rank, 16 Threads/ Rank)

```
export OMP_NUM_THREADS=16
gridFileName=grid1MPI_3x3x4.cmg  #this file is included.
Order=16
Groups=200
quadType=2
Polar=9
Azim=10
srun -n 1 -N 1 \
    ../Teton/SuOlsonTest \
    $gridFileName $Groups $quadType $Order $Polar $Azim
```

7.6.2 Initial Crossroads Example 2 (4 Nodes, 64 MPI Ranks, 1 Thread/ Rank)

```
export OMP_NUM_THREADS=1
gridFileName=grid64MPI_3x3x4.cmg  # this file is included.
Order=16
Groups=200
quadType=2
```

```
Polar=9
Azim=10
srun -n 64 -N 4 \
    ../Teton/SuOlsonTest \
    $gridFileName $Groups $quadType $Order $Polar $Azim
```

7.6.3 Initial Crossroads Example 3 (1,944 Nodes, 46,656 MPI Ranks, 1 Thread/Rank)

This is intended for a cluster with 2x 12-core Intel Ivy Bridge processors.

```
export OMP_NUM_THREADS=1
gridFileName=grid46656MPI_3x3x4.cmg   # this file is included.
Order=16
Groups=200
quadType=2
Polar=9
Azim=10
srun -n 46656 -N 1944 \
    ../Teton/SuOlsonTest \
    $gridFileName $Groups $quadType $Order $Polar $Azim
```

7.6.4 Crossroads Baseline Small Case (1 Node, 1 MPI Rank, 32 Threads/Rank)

This case was run on Trinity, utilizes approx. 2.336 GB total memory, and achieved a figure of merit (FOM) value of 2.90956e+08 unknowns per second (from 27 cumulative iterations and a work time of 11.3317 seconds).

```
export OMP_NUM_THREADS=32
gridFileName=grid1MPI_4x4x4.cmg   # this file is included.
Order=16
Groups=200
quadType=2
Polar=9
Azim=10
srun -n 1 -N 1 \
    ../Teton/SuOlsonTest \
    $gridFileName $Groups $quadType $Order $Polar $Azim
```

7.6.5 Crossroads Baseline Medium Case (64 Nodes, 2,048 MPI Ranks, 1 Thread/Rank)

This case was run on Trinity, utilizes approx. 5.506 TB total memory, and achieved a figure of merit (FOM) value of 4.13968e+10 unknowns per second (from 38 cumulative iterations and a work time of 229.565 seconds).

```
export OMP_NUM_THREADS=1
gridFileName=grid2048MPI_4x4x4.cmg   # this file is included.
Order=16
Groups=200
quadType=2
Polar=9
Azim=10
srun -n 1 -N 1 \
    ../Teton/SuOlsonTest \
    $gridFileName $Groups $quadType $Order $Polar $Azim
```

7.6.6 Crossroads Baseline Large Case (3,906 Nodes, 125,000 MPI Ranks, 1 Thread/ Rank)

This case was run on Trinity, utilizes approx. **336.088 TB** total memory, and achieves a figure of merit (FOM) value of **1,370,710,000,000 unknowns per second** (from 71 cumulative iterations and a work time of 790.644 seconds).

```
export OMP_NUM_THREADS=1
gridFileName=grid125000MPI_4x4x4.cmg   # this file is included.
Order=16
Groups=200
quadType=2
Polar=9
Azim=10
srun -n 1 -N 1 \
    ../Teton/SuOlsonTest \
    $gridFileName $Groups $quadType $Order $Polar $Azim
```

8 UMT Figure of Merit

The performance of platforms is tested with a weak scaling experiment. A Figure of Merit (FOM) will be reported. This is done in `main()` and can be perused in the file `SuOlson-Test.cc`. The FOM is:

```
number_Unknowns/cumulativeWorkTime*cumulativeIterationCount
```

The FOM will grow as more and more MPI ranks are added and as more and more OpenMP threads are added. There is more on executing UMT2015 below.

9 UMT Run Rules

The following enumerated list contains the overall run rules.

1. **Baseline Run** - UMT must be run without modification or substitution of existing routines by optimized libraries. Aggressive compiler and OpenMP optimizations may be used as long as the compiler and optimizations are generally available and fully supported, the source is not otherwise altered, and the benchmark verifies correct operation.

2. Benchmark runs must be performed with the benchmark reporting total memory for data of at least 344,000 GB.

10 References

1. P.F. Nowak and M.K. Nemanic, “Radiation Transport Calculations on Unstructured Grids Using a Spatially Decomposed and Threaded Algorithm,” Proc. Int. Conf. Mathematics and Computation, Reactor Physics and Environmental Analysis in Nuclear Applications, Madrid, Spain, September 27-30, 1999, Vol. 1, p. 379 (1999).

2. Paul Nowak, “Deterministic Methods for Radiation Transport: Lessons Learned and Future Directions”, ASC Workshop on Methods for Computational Physics and Modern Software Practices, March 2004, LLNL document UCRL-CONF-202912.