Demonstrating the Impact of OpenMP Overheads in Multi-Physics Using a Mini App

Dylan McKinney, Ian Karlin, Riyaz Haque
Lawrence Livermore National Lab

Abstract

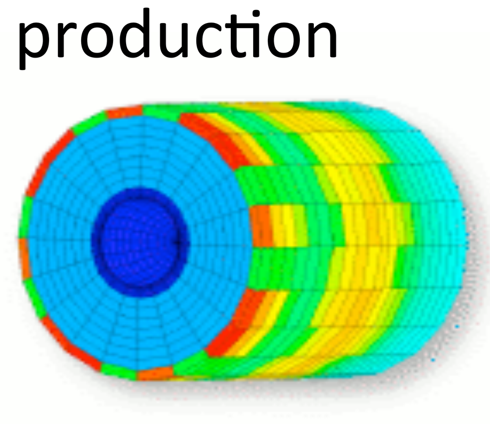
Pure MPI programming approaches only strong scale so far due to surface to volume issues. Switching to an MPI+OpenMP programming model can exploit growing concurrency while minimizing surface to volume challenges. However, for small problem sizes, the amount of work within an OpenMP region may not amortize thread overheads. KULL, a multi-physics production code, has this problem. In this poster, we demonstrate that large OpenMP overheads limit the thread scaling of KULL. We show that the Lightweight OpenMP (LOMP) compiler reduces overheads enough to allow KULL to scale further. We duplicate this issue in LULESH, a hydrodynamics proxy application. LULESH, at small problem sizes, sees up to 5x speed-up using LOMP. By demonstrating the impact overheads have on small problems, and duplicating this issue in a proxy application, we indicate to vendors that OpenMP overhead limits performance and provide them with benchmark test problems that show our OpenMP challenges.

Goal

- Demonstrate OpenMP overhead concerns of KULL using LULESH

KULL

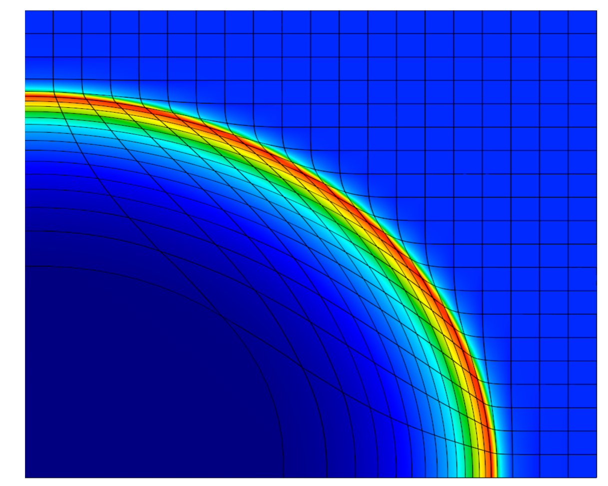
- High Energy Density Multi-Physics production app
  - Massively parallel
  - Multi-Physics Rad-Hydro Code
  - Models 3D Sn Radiation coupled to hydro
    - Sn has 3 extra dimensions modeled: angle (2) and energy
    - Dominates memory and total runtime



All tests run on one BG/Q node. 16 GB mem and 16 cores per node. 4 hardware threads per core

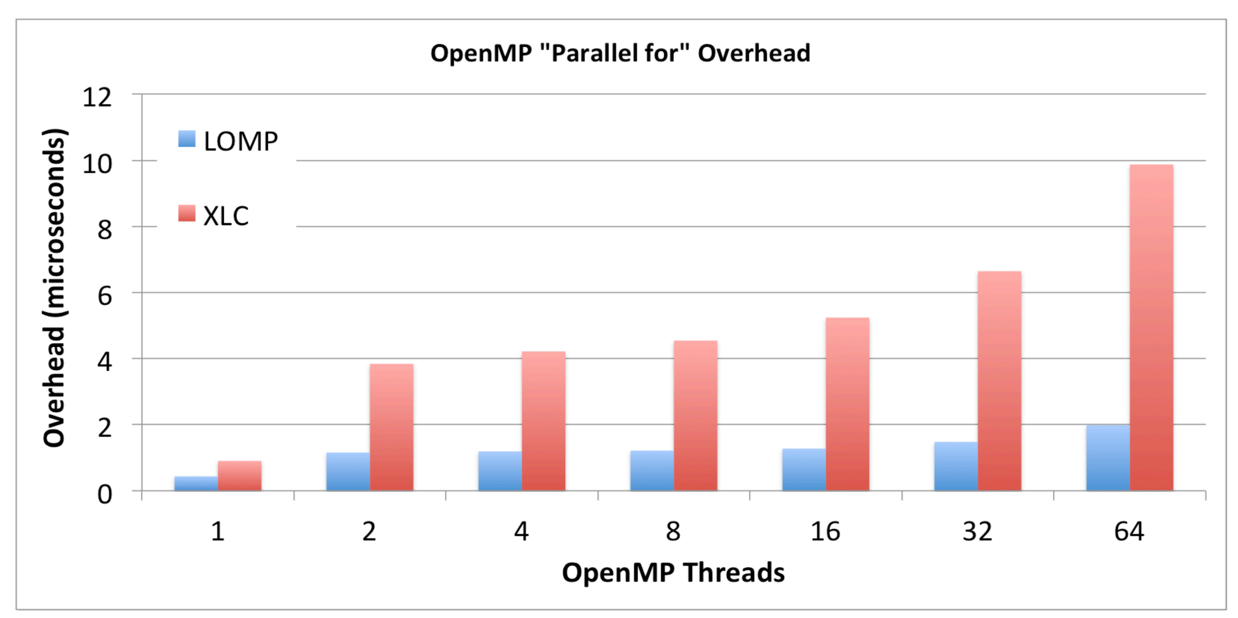
LULESH[1]

- Shock-Hydro proxy app
  - Darpa UHPC problem
  - Solves Sedov blast
  - Represents production hydrodynamics



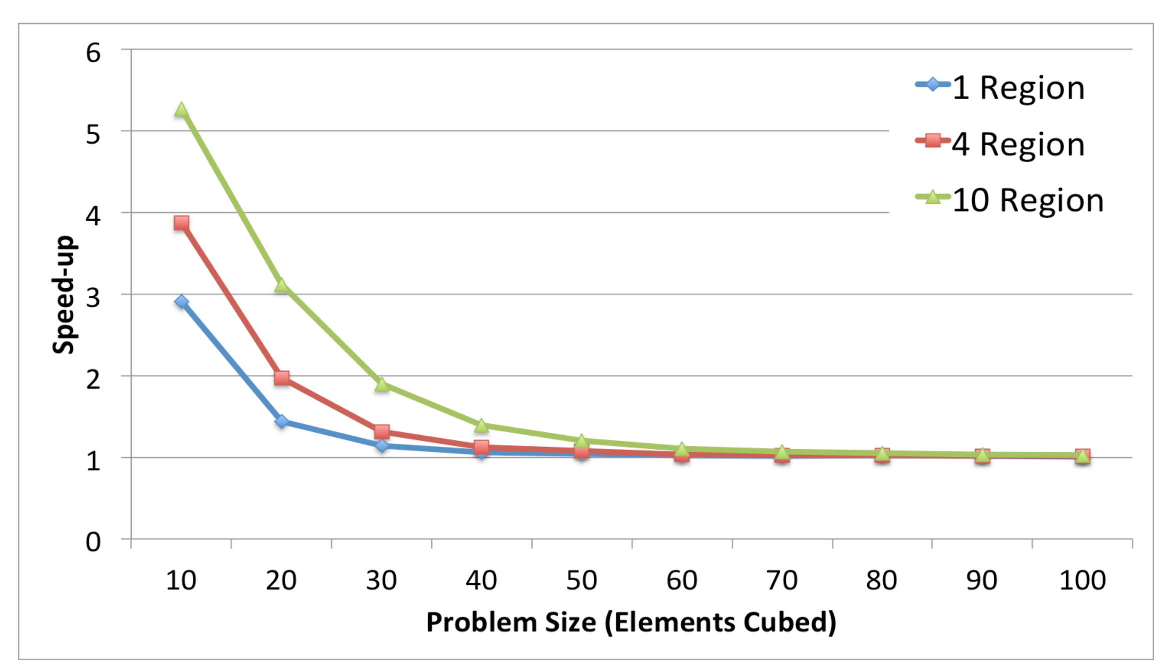
OpenMP Overheads

- Overhead associated with thread spawning and synchronization
- Lightweight OpenMP beta compiler (LOMP)[2] reduces these
  - Overheads 2-5x less than default compiler



LOMP and Default (XLC) overhead comparison

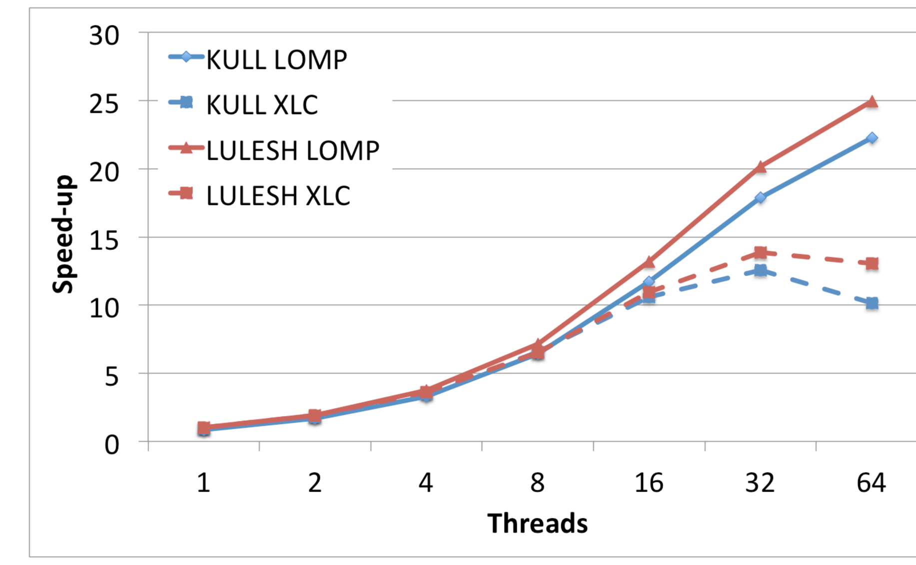
Overhead Impact as problem size varies LULESH (64 threads)



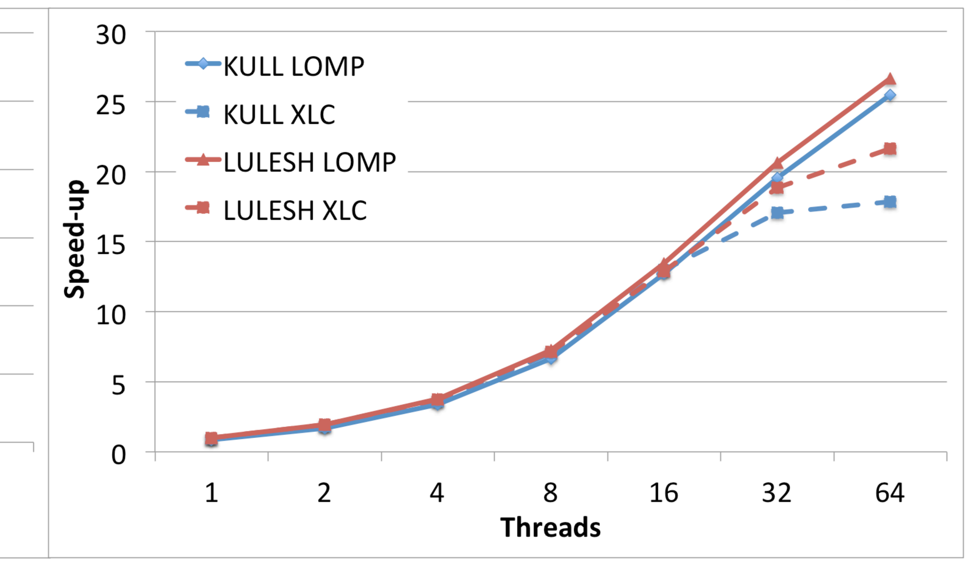
Not enough parallel work to amortize overheads at small sizes

Results

8,000 Elements 4 Regions



32,000 Elements 4 Regions

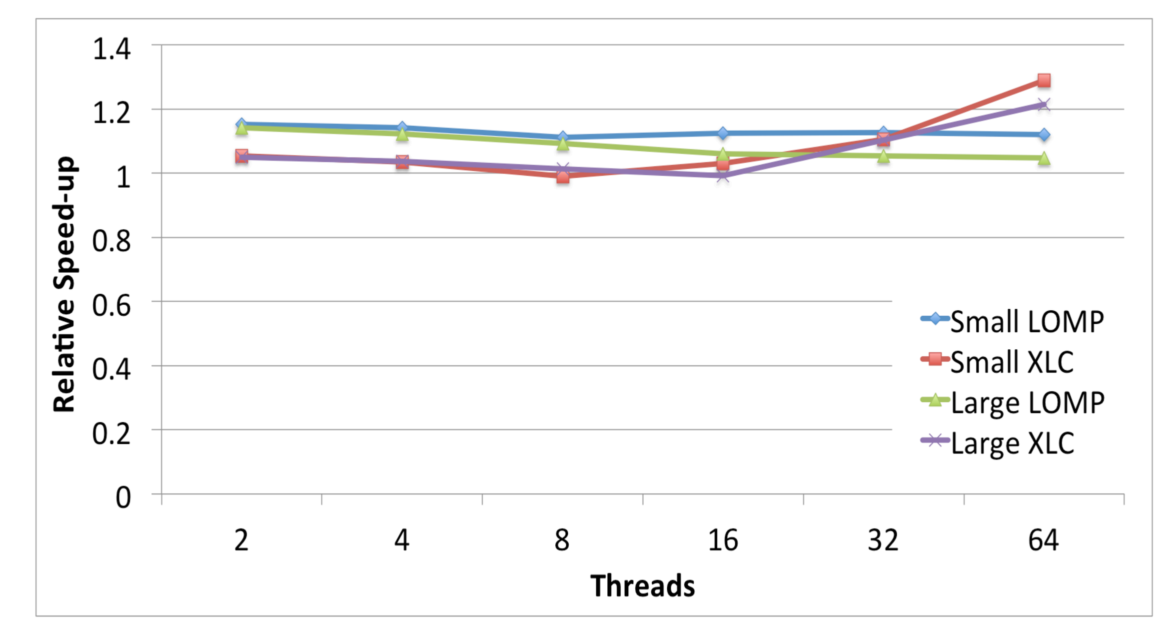


Runtimes (s)
8,000 Elements

App/Compiler	LULESH XLC	LULESH LOMP	KULL XLC	KULL LOMP
Serial	6.26	6.34	89.18	89.18
1 Thread	6.61	6.41	103.92	96.29
Best	0.25	0.47	7.09	4.00

LULESH and KULL thread scaling with LOMP and XLC

Correspondence:
1 : Perfect Match
1 > : KULL Speed up Greater
1 < : KULL Speed up Less



LULESH Speed-Up normalized to KULL.

Conclusion and Future Work

In this poster we demonstrated that OpenMP overheads impact the scaling of LULESH similarly to KULL for various problem sizes and region counts. We showed that for small problem sizes, in both applications, small OpenMP overheads are crucial to performance. In the future we plan to leverage these results to demonstrate to vendors why low OpenMP overheads are important to KULL runtime performance. Our data also shows how they can reproduce this behavior and test their own runtimes using LULESH.

References

1. Hydrodynamics Challenge Problem Lawrence Livermore National Laboratory. LLNL-TR-490254. Livermore, CA.
2. A. E. Eichenberger and K. O'Brien, "Experimenting with low-overhead openmp runtime on ibm blue gene/q," *IBM Journal of Research and Development*, vol. 57, no. 1/2, pp. 8:1–8:8, Jan 2013.

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC. LLNL-POST-698273