Exploring Asynchronous Many-Task Runtime Systems toward Extreme Scales

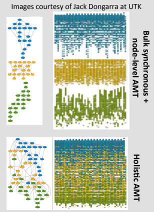
Samuel Knight (Florida Institute of Technology), Marc Gamell (Rutgers University), Gavin Baker (California Polytechnic State University), David Hollman (Sandia National Labs), Gregory Sjaardema (Sandia National Labs), Hemanth Kolla (Sandia National Labs), Keita Teranishi (Sandia National Labs), Jeremiah Wilke (Sandia National Labs), Nicole Slattengren (Sandia National Labs), Janine C. Bennett (Sandia National Labs)

Introduction

Asynchronous many-task (AMT) runtime solutions

- Show promise at sustaining performance in spite of system performance heterogeneity
- Task-graph
- Nodes are work (tasks) are data dependencies
  - Active area of research
    - Charm++, PaRSEC, HPX, Legion, OCR, STAPL, Uintah, StarPU, Swift/T

Images courtesy of Jack Dongarra at UTK



Comparative Analysis of AMT Runtime

Asynchronous many-task programming models are a leading new paradigm with many variants

Goal: Address knowledge gaps

Comparative analysis of leading candidate solutions

Quantitative & qualitative tests using ASC-relevant codes

Outcome: Guidance to code development road map for next generation platforms for ASC/Integrated Codes

We selected Charm++, Legion and Uintah

Demonstrated science applications at scale

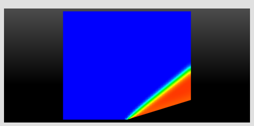
Maturity of runtime

Three very different implementations, APIs, sets of abstractions

Accessibility of teams

Target Application: MiniAero

- 3-dimensional, finite volume, CFD
- Runge-Kutta fourth order time marching
- Options for 1st or 2nd order spatial discretization
- Inviscid Roe and Newtonian fluxes
- Baseline 3800 lines C++ (MPI +Kokkos)



$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_j}{\partial x_j} = 0 \quad \text{Mass}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_i u_j + P\delta_{ij}\right) = \frac{\partial \tau_{ij}}{\partial x_j} \quad \text{Momentum}$$

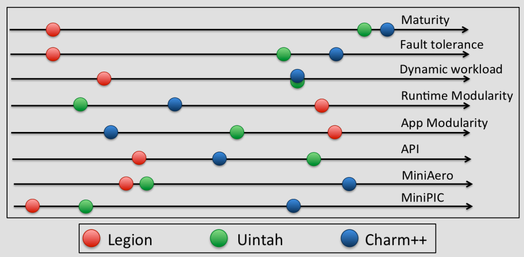
$$\frac{\partial \rho E}{\partial t} + \frac{\partial \rho u_j H}{\partial x_j} = -\frac{\partial q_j}{\partial x_j} + \frac{\partial u_i \tau_{ij}}{\partial x_j} \quad \text{Energy}$$

Qualitative Evaluation

Programmability: Does this solution enable expression of our workloads?
Mutability: Ease of adopting this solution, modifying it to suit our needs?

Qualitative measures can help assess programmability and mutability	
Maturity	How stable is the API? Likelihood of encountering bugs?
Fault tolerance support	What fault models/recovery mechanisms are supported? How easy is the API?
Dynamic workload support	What load-balancing/work stealing mechanisms are supported? How easy is the API?
Modularity (Runtime and Application)	How reusable are components? How extensible is the framework?
API	What do developers like/dislike regarding the interface?
MiniAero	How easy is it to express MiniAero?
MiniPIC	How easy is it to express MiniPIC (Particles in Cell)?

Our internal survey indicates that there is no runtime solution that addresses all of our needs from qualitative (and subjective) perspectives

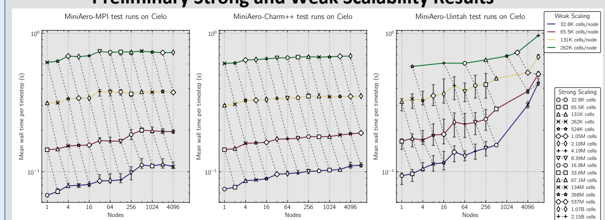


Summary of Qualitative Evaluation

Runtime	Key strengths	Key Weaknesses from Sandia's application perspective	Compelling near-term leverage points
Legion	On-node logical/physical dependency model	Rigidity of data model, doesn't support dynamic data fetching/push model	High-level runtime/API facilitates specific use cases; REALM is full-featured event runtime
Charm++	Flexible data movement patterns, supports push model	Lack of data model, template support, logical work regions (for multi-dimensional load balancing), CI file interface	Mature back-end and API for initial algorithmic exploration of specific use cases
Uintah	Application-driven approach very performant for specific-use cases	Lack of support for unstructured meshes	Application APIs and abstractions

Quantitative Evaluation

Preliminary Strong and Weak Scalability Results

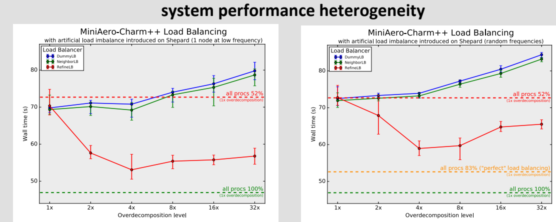


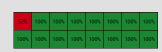
- 8192 nodes of Cielo Cray XE6 @ Sandia/LANL
  - 2 8-core CPUs per node with 32GB memory
  - 3D Torus (16 x 12 x 24)
  - 6.57 x 4.38 x 4.38 TB peak network bandwidth

- FlatMPI code
- 2X over-decomposition with Charm++
- No over-decomposition with Uintah
- No SMP-based implementation for Charm++ and Uintah
- Runtime issues with Legion.

- Very competitive to MPI
- Charm++ outperforms MPI
- Uintah has some issues with small problem sizes
  - (Recent result with Titan indicates ideal scalability for 1B+ cells)

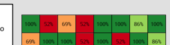
Preliminary studies show an AMT RTS can mitigate system performance heterogeneity





System imbalance

- 16 nodes on Shepard @ Sandia
  - 2 16-core Intel Haswell CPUs per node
  - System allows frequency scaling from 1.2 to 2.3GHz
- 15 time steps
- Load balancing each time step
- 4.1 million cells/node



System imbalance

Conclusions

Pros:

- AMT runtimes show tremendous potential for addressing extreme-scale challenges
- The collective research being performed by AMT RTS developers are critically important precursor to establishing community standards

Cons:

- All leading AMT programming models and RTS have been designed or demonstrated on a limited set of applications
- None of the runtimes appear to satisfy all requirements of our application workloads (definition of requirements is still in progress)
- None of the runtimes are production ready for a broad class of applications

Acknowledgements

- National Energy Research Scientific Computing Center
- Alex Aiken, Wonchan Lee and Elliot Slaughter (Stanford University)
- Michael Bauer and Sean Treichier (NVIIDA)
- Nikhil Jain, Laxmikant Kale and Eric Mikida (University of Illinois, Urbana-Champaign)
- Martin Berzins, Todd Harman, and Alan Humphrey (University of Utah)
- Robert Clay, David DeBonis, Ryan Grant, Simon Hammond, Victor Kuhns and Stephen Olivier (SNL)

References

[1] M. Bauer, S. Treichler, E. Slaughter, and A. Aiken. Legion: Expressing locality and independence with logical regions. In Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, SC '12, pages 66:1{66:11, Los Alamitos, CA, USA, 2012. IEEE Computer Society Press.

[2] M. Berzins, J. Schmidt, Q. Meng, and A. Humphrey. Past, present and future scalability of the uintah software. In Proceedings of the Extreme Scaling Workshop, BW-XSEDE '12, pages 6:1{6:6, Champaign, IL, USA, 2012. University of Illinois at Urbana-Champaign.

[3] L. V. Kale and A. Bhatele, editors. Parallel Science and Engineering Applications: The Charm++ Approach. Taylor & Francis Group, CRC Press, Nov. 2013.

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.
SAND No. 2015-XXXXP