High Performance Workflows for Molecular Dynamics Simulations

Justin M. Wozniak <wozniak@mcs.anl.gov>,[12] Andreas Wilke,[1] Nicola Ferrier,[12]
Daniel Reid,[2] and Sidney Nagel[2]
1: Argonne National Laboratory 2: University of Chicago

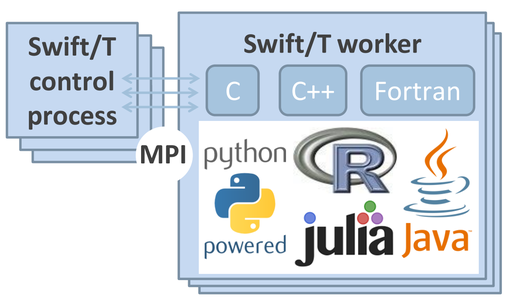
http://swift-lang.org

Overview: Dataflow programming at scale

- Many important application classes that are driving the requirements for extreme-scale systems can be elegantly expressed as many-task data flow programs:
  - ✔ Optimization
  - ✔ Stochastic programming
  - ✔ Automatic classification
  - ✔ Uncertainty quantification
- The data flow programming model of the Swift parallel scripting language can elegantly express the massive concurrency demanded by these applications through implicit parallelism, which has the productivity benefits of a high-level language [1,2]
- Here, we applied the Swift/T language implementation and runtime to an example problem relevant to MICCoM
- Swift/K was previously implemented as a workflow language (c. 2007) for distributed computing (grids, clouds, etc.). The current implementation (Swift/T) is designed for HPC

Dataflow processing in distributed memory

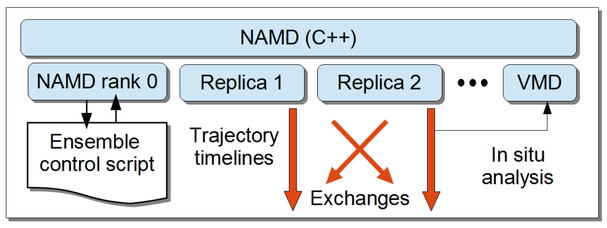
- Swift/T allows users to bundle many different application codes - in many different languages – into one in-memory workflow [3]
- The Swift/T programming model allows data dependent execution of tasks written in these languages, making up a high-performance workflow
- State (including threads) can be left on a worker and accessed later



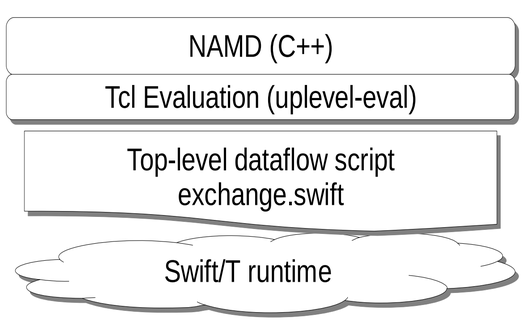
- Swift/T workflows can contain trillions of tasks, executing billions per second on systems like the IBM Blue Gene/Q or large Cray installations
- Tasks can use MPI internally

Prior work: Replica Exchange with NAMD

- Collaborative effort to improve the replica exchange scripts used by the NAMD team at UIUC [4]
- Desired more flexibility with use of MPI, better automatic load balancing, richer programming model
- Needed support for scripting interfaces to native code (C++), inversion of control, high performance and portability to NCSA Blue Waters

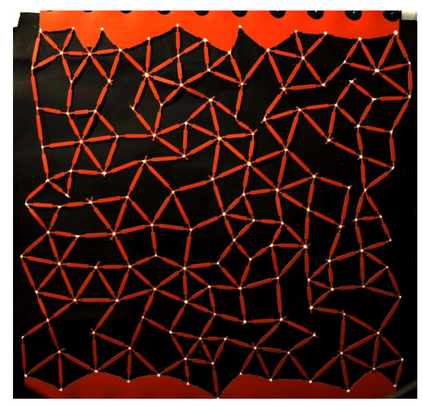


Swift job



Auxetic workflow collaboration

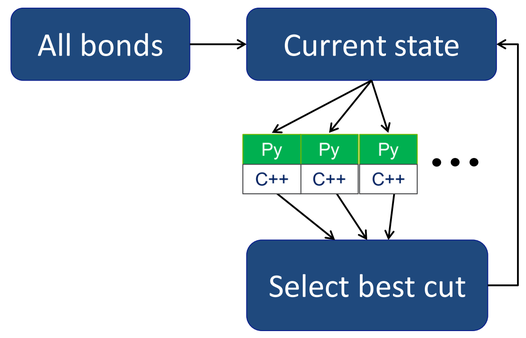
- New Swift/T workflow to optimize materials design problem
- Considers a network of particles connected by bonds which can be selected and cut to optimize the material response under stress
- Since any combination of bonds may be cut, there is an extremely large search space (limited only by physical constraints)
- Each Auxetic task is a call to a C++ function that measures the impact of a given cut on the system
  - For a 300 particle test system, this takes ~1.8s on Midway (UofC cluster); planning for 2000 particles @ 2 minutes
- The C++ functionality is exposed via Python interfaces (Boost.Python)
- A Swift workflow orchestrates the search and distributes work across Midway
- The current workflow has been tested on Midway up to 360 cores



Example system

Status and future plans

- Exploring search strategies, control cases for comparison:
  - Brute force
  - Evolutionary algorithms
- Current search evaluates all possible single cuts from a starting point, picks the best, and iterates from there
- Evaluating more than one cut at a time will provide more concurrency



Workflow diagram

- Port to larger machine such as UofC Cray Beagle (should be straightforward)

References

1. **Swift/T: Large-scale application composition via distributed-memory data flow processing** J. M. Wozniak, T. G. Armstrong, M. Wilde, D. S. Katz, E. Lusk, I. T. Foster. Proc. CCGrid 2013.
2. **Compiler techniques for massively scalable implicit task parallelism** Timothy G. Armstrong, Justin M. Wozniak, Michael Wilde, and Ian T. Foster. Proc. SC 2014.
3. **Interlanguage parallel scripting for distributed-memory scientific computing** Justin M. Wozniak, Timothy G. Armstrong, Ketan C. Maheshwari, Daniel S. Katz, Michael Wilde, and Ian T. Foster. Proc. WORKS 2015.
4. **Petascale Tcl with NAMD, VMD, and Swift/T** James C. Phillips, John E. Stone, Kirby L. Vandivort, Timothy G. Armstrong, Justin M. Wozniak, Michael Wilde, and Klaus Schulten. Proc. High Performance Technical Computing in Dynamic Languages at SC 2014.

Midwest Center for Computational Materials (MICCoM) Scientific Advisory Board Meeting, October 18 & 19, 2016