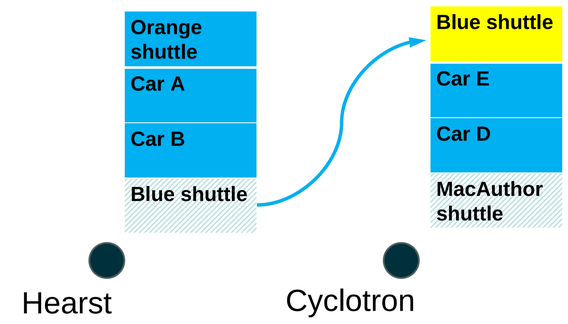
Benchmarking Devastator: A Novel Parallel Discrete Event Simulator Using a Hybrid Communication Model

Jianlan Ye[1,2], Tan Nguyen[2], Maximilian Bremer[2], John Shalf[2]
[1]Arizona State University, [2]Lawrence Berkeley National Laboratory

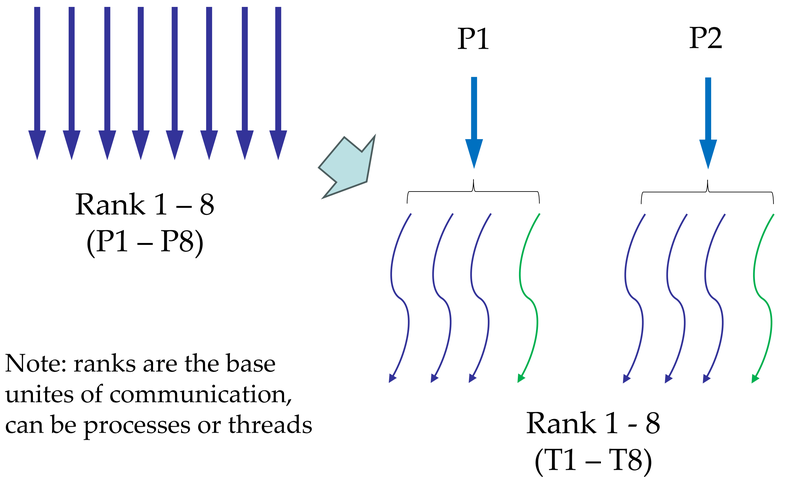
What are Parallel Discrete Event Simulations

- Simulations that partition the problem domain into discretized logical processes (LP) with state variables
- LPs execute events asynchronously, and pass timestamped events to themselves and other LPs
- Events MUST be executed in chronological order in each LP, otherwise rollbacks are necessary
- Applications include traffic simulations, fluid simulation, etc.



What is New

- Light-weight thread communication was added to take advantage of the local communications
- Asynchronous global virtual time (GVT) computation
- Easier to use with modern C++ and self-adaptable synchronization



Note: ranks are the base unites of communication, can be processes or threads

Single-node Scaling

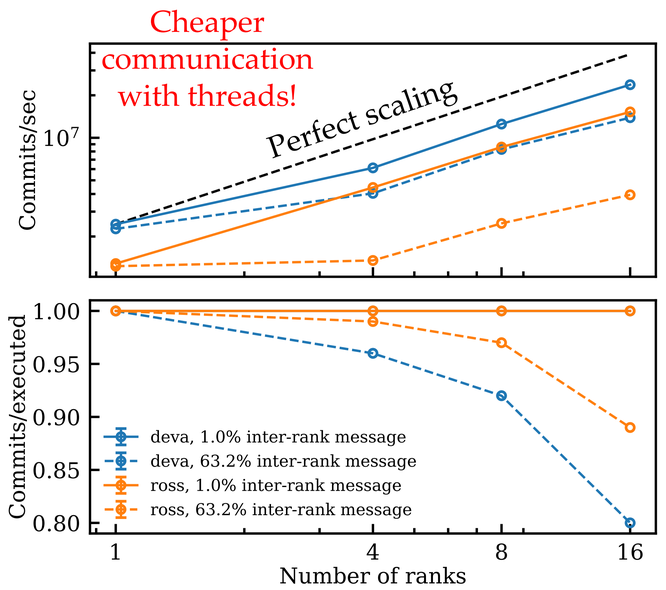


Fig. 1 Single-node scaling for devastator and ROSS with different communication pattern. Devastator has less throughput reduction when has more inter-rank communications

Threads vs. Processes

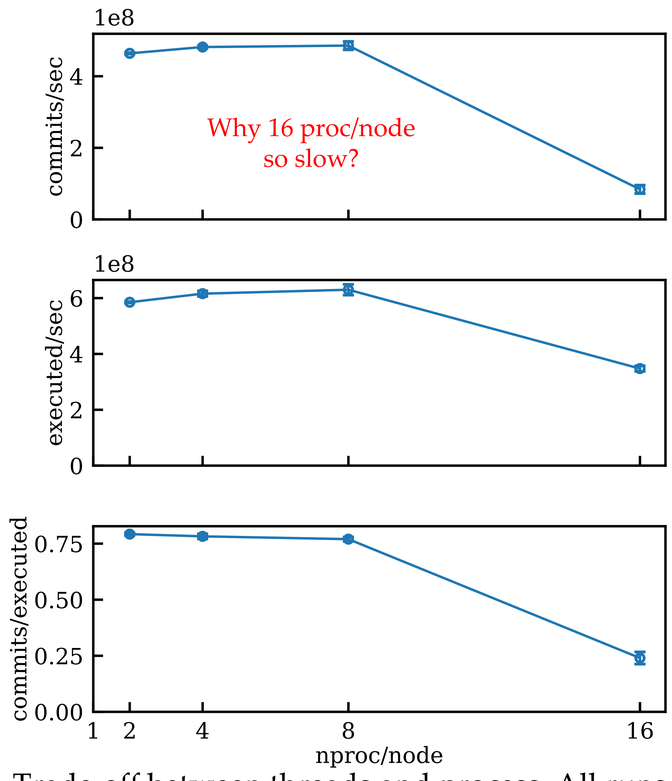


Fig. 2 Trade-off between threads and process. All runs were conducted on 4 nodes with fixed problem size (128 cores per node) while varying the processes per node, i.e. different threads per process

Time Breakdown

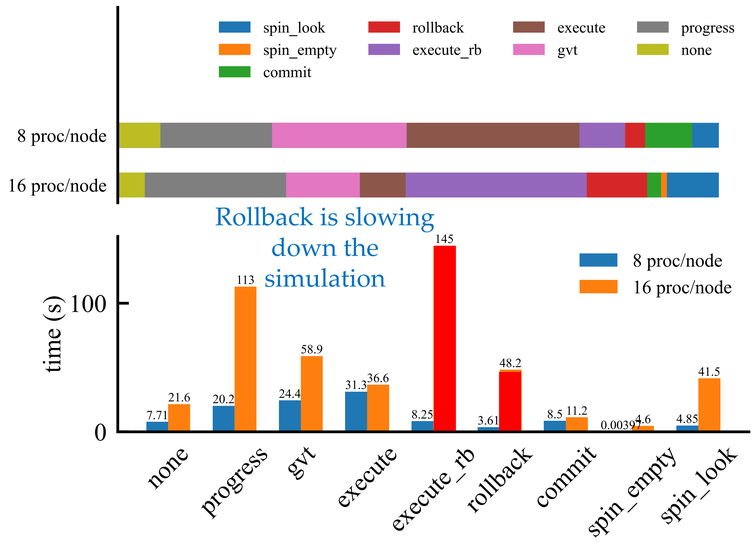


Fig. 3 Time breakdown of the 8 proc/node and 16 proc/node runs. Rollbacks and rollback executes for the 16 proc/node takes disproportionally longer time than the 8 proc/node case.

Multi-node Scaling

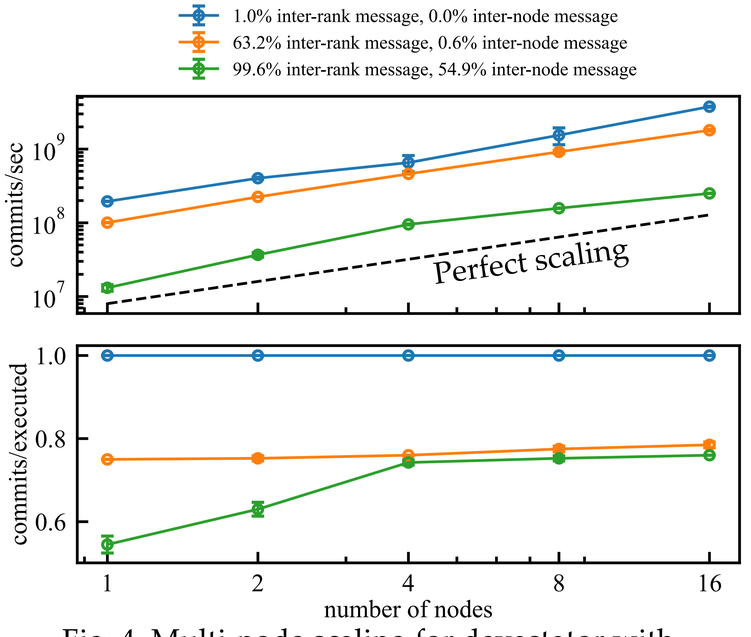


Fig. 4 Multi-node scaling for devastator with different communication pattern

Due to the heavily concurrent nature of PDES, we are facing issues on Perlmutter. Please let us know if you have any insights