2020-21 TST Team Visit: Center Overview

Prof. Patrick G. Bridges University of New Mexico





Center for Understandable, Performant Exascale Communication Systems

Overview

- Introductions and new project personnel
- Administrative Updates
- Overall goals and approach
- Brief summary of major activities
- Current challenges/issues





Schedule after this talk

- This morning discussion of major activities
 - Overview (this talk)
 - Assessment planning and next steps (Puri)
 - Initial Assessment Results (Me)
 - Optimization/Implementation (Tony and Derek)
- Lunch/Early Afternoon Research talks
 - Amanda Bienz (UNM)
 - Student lightning talks
- Afternoon Time for Additional discussions
- Schedule is flexible can change as needed





Introductions

- TST Team Members
 - Jon Riesner (LANL)
 - Kevin Pedretti (SNL)
 - Olga Pearce (LLNL)
- Other DOE Personnel
- Center Senior Personnel
 - Patrick Bridges (UNM)
 - Puri Bangalore (UAB)
 - Tony Skjellum (UTC)
 - Abi Arabshahi (UTC)
 - Amanda Bienz (UNM)
- Administrative Support
 - Tracy Wenzl (UNM)

- Postdocs
 - Thomas Hines (UTC)
 - Ryan Marshall (UTC)
 - (*) Derek Schafer (UTC)
- Students
 - Gerald Collom (UNM)
 - Jered Dominguez-Trujillo (UNM)
 - Keira Haskins (UNM)
 - (*) Pepper Marts (UNM)
 - Tanner Broaddus (UTC)
 - Thomas Gorham (UTC)
 - (*) Garrett Hooten (UTC)
 - (*) Carson Woods (UTC)

(*) Supported by other DOE funds





Administrative update

- All contracts and subcontracts in place
- SARAPE and system access moving smoothly
- Student spring/summer placements proceeding
 - Jered Dominguez-Trujillo Sandia (spring 2020)
 - Keira Haskins Sandia
 - Gerald Collom LLNL
 - Others TBD





Overall Goal: Performance Portability

- Data movement in complex systems (accelerators, memory hierarchies, NICs) limit application performance
- The communication system should be in charge of optimizing data movement between processes/nodes
- For that to happen, communication abstractions must:
 - Supply the information needed to optimize data movement
 - Not overly-constrain application programmers, frameworks, or runtimes (e.g. synchronization)
 - Actually be efficient and easy to use
- Need usable communication abstractions that capture applications' *high-level communication plans*

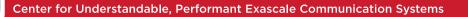




What happens today?

- Most Applications and frameworks use the low-level communication primitives in MPI
 - Isend, Irecv
 - Primitive types
 - Hand-packed buffers
 - Global synchronous collectives
- Because they:
 - Perform predictably across systems
 - Are consistently well optimized
- And these work as well as application programming in C and assembly has always worked





What could be done?

- MPI has higher-level abstractions that *could* help
 - In the standard (derived data types, neighbor collectives, persistent communication)
 - In upcoming standards (partitioned communication)
 - In development (combinations of these features)
- Why aren't they used?
 - Real applications don't use abstractions that are not consistently well-optimized across implementations
 - Developers and vendors don't optimize abstractions that aren't used in real applications
- There are also still big gaps in the abstractions available





Example

```
    Cabana Distributor class pseudocode
```

```
for ( int n = 0; n < num_n; ++n ) {
    auto recv_subview = Kokkos::subview( recv_buffer, recv_range );
    MPI_Irecv( recv_subview.data(), MPI_BYTE, distributor.neighborRank( n ));
}
for ( int n = 0; n < num_n; ++n ) {
    auto send_subview = Kokkos::subview( send_buffer, send_range );
    MPI_Send( send_subview.data(), MPI_BYTE, distributor.neighborRank( n ));
}
MPI_Waitall();
Kokkos::parallel_for( "Cabana::Impl::distributeData::extract_recv_buffer",
    extract recv buffer policy, extract recv buffer func );</pre>
```

- This is literally a hand-built neighbor collective on a graph communicator
- Kokkos code packs/unpacks complex types without knowledge of the network or communication system
- But most MPIs don't optimize this yet, particularly with complex types, so:

This is the right thing for programmers to do today even though it cripples MPI's ability to do better





High-Level Approach

- Understand the *desired* mapping from the application to the communication system
 - What and how do *applications* want to communicate?
 - What can communication systems effectively optimize?
- Create abstractions that bridge this divide well
 - What communication abstractions should exist to support applications well?
 - How should we optimize these abstractions so that they perform consistently and efficiently?
- Complements research on optimizing *existing* abstractions by other research groups





Brief Summary of Major Activities to date

- Formative Assessment
 - Initial examination of key mini-apps in collaboration with TST team members and lab collaborators
 - Full draft assessment plan, including production apps and associated mini-applications
 - Assessment of application tracing tools
 - Collection and initial examination of existing application traces and tools
 - Work on getting access to DOE production codes
 - Design of scalable quantitative analysis approach





Brief Summary of Major Activities to date

- Research Infrastructure
 - Assessment of ExaMPI needs for supporting initial mini/proxy application set
 - Implementation of additional ExaMPI primitives
 - Repository for holding application traces
 - Initial work designing/adopting experiment management infrastructure





Brief Summary of Major Activities to date

- Implementation/Modeling/Optimization
 - Initial evaluation of neighbor collectives in a miniapplication
 - Partitioned communication implementation and measurement
 - Modeling variation in application performance
 - GPU support for partitioned communication
 - Modern C++ interface for performance





Outstanding Issues/Challenges: General assessment

- Appropriateness of chosen applications and proxies for assessment
- Prioritization feedback for final set of assessed applications
- Appropriate multi-scale/multi-physics proxy
 - Examining UCDavis Z-Model as potential example
 - Would need to develop new code/codes





Outstanding Issues/Challenges: Codes and inputs for assessment

- Collaborations with lab partners have helped identify proxy and miniapps that reasonably characterize production applications
- Current access to production codes for assessment
 - HIGRAD_basic (EAR 99): Access at UNM/LLNL in progress with LANL
 - EMPIRE (ITAR): Access on SNL restricted systems in progress
 - EMPIRELite (EAR99): Still in development at SNL
- Would still like to directly qualitatively and ideally quantitatively largescale applications, either via additional traces or by direct measurement
- Some assessment can happen on summer internships, but would ideally like to limit reliance on these
- Need (ideally unrestricted) input decks for export-controlled applications that expose relevant communication problems and can be easily published





Outstanding issues/challenges: Access to test platforms

- Current systems in used
 - Large-scale: Lassen, Quartz, Solo, Stria, Capulin
 - Testbed: Some SNL ECN systems
- Have access to:
 - GPUS: NVIDIA V100 (Lassen, UNM)
 - IBM/Intel/ARM CPUs
 - Mellanox, Intel, Cray NICs
- Will need access testbed access to AMD and Intel GPU systems





Questions/Additional Discussion?



Center for Understandable, Performant Exascale Communication Systems

