Recent InfiniBand Optimizations in Chapel

Elliot Ronaghan
Hewlett Packard Enterprise
USA
elliot.ronaghan@hpe.com

***Abstract*—This talk will highlight recent optimizations that have significantly improved Chapel's performance and scalability on InfiniBand systems. Enhancements to the memory registration implementation have improved the performance of several core benchmarks and user applications including Arkouda, a Python package backed by Chapel that provides a key subset of the NumPy and Pandas interfaces. Performance results for core benchmarks will be shown on a small scale InfiniBand cluster and Arkouda results will be shown on a 240 node InfiniBand-based HPE Apollo system.**

I. Introduction

Historically, we have focused on optimizing Chapel's performance on Cray networks with the intention of ensuring we had the right language features first and then optimizing for other networks. Over the last few Chapel releases we have started optimizing more for InfiniBand networks, which we target using the GASNet communication library. With recent memory registration optimizations, applications can now achieve better performance on a modern InfiniBand network than they can on a Cray Aries network.

II. Memory Registration

On most high performance networks, including InfiniBand, memory has to be registered with the network in order to do one-sided communication. This support for one-sided communication or Remote Dynamic Memory Access (RDMA) is critical for achieving good performance in Chapel.

On InfiniBand, Chapel uses GASNet's dynamic registration mode where memory is registered with the network at communication time. This results in fast startup times and good NUMA affinity, but can incur a performance penalty to RDMA operations because memory has to be registered the first time it is communicated. This cost is typically amortized over multiple RDMA operations, but doing so requires tracking which regions of memory have already been registered.

Bottlenecks in this registration tracking code previously limited performance and scalability. We collaborated with the GASNet team to resolve these performance issues, which will be detailed in this talk.

III. Performance Results

Performance results for core benchmarks and user applications will be shown on a 16 node InfiniBand cluster that is used for Chapel's nightly performance testing. Example improvements include a 4x speedup for ISx, an integer sort benchmark, as well as 4x improvement for NAS Parallel Benchmarks FT.

Additionally, performance results for Arkouda will be shown on a 240 node InfiniBand-based HPE Apollo system. The following figures summarize the Arkouda results.

Figure 1 shows Arkouda argsort performance on 240 nodes (30K cores) of an HPE Apollo system with an HDR-100 InfiniBand network. Registration tracking issues caused performance to fall off dramatically after 64 nodes. Fixing the tracking issues as well as implementing application optimizations enabled by that fix result in a 50% speedup at 16 nodes, and nearly a 20x speedup at 240 nodes.

Figure 2 shows argsort performance on 240 nodes of the same HPE Apollo system compared to a Cray XC with an Aries network. With an equivalent problem size of 8 GiB per node, Apollo performance is 50% ahead of XC at 240 nodes. Apollo systems can have much higher memory capacities, enabling larger problem sizes. With 64 GiB per node, 240 node Apollo performance is nearly 3x better than the XC.

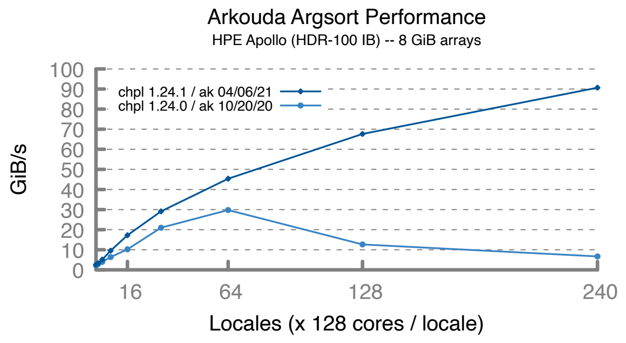

Fig. 1. Argsort scalability before and after registration improvements

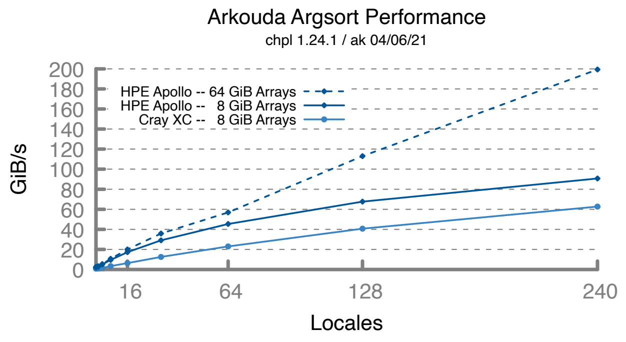

Fig. 2. Argsort scalability comparing Cray XC to HPE Apollo