The objective of this project is the creation of ReFrame regression tests that measure intranode and internode latency and bandwidth. The tests will use OSU Micro-Benchmarks, a set of utilities designed to test the communication performance of a cluster for various parallel programming frameworks. This project will focus on testing MPI communication primitives.

Test design

The 2 main tests in OSU Micro-Benchmarks that evaluate communication latency and bandwidth are

- osu_latency, and
- osu_bw

respectively. By default, the benchmarks evaluate the performance of the system fo a set of messages of different length, as the length of the message affects both the latency and the throughput of the communication. You can find more details in the documentation MPI tests [1].

To extract meaningful conclusions from the test results, you should consider the architecture of the underlying system. Broadly speaking, there following cases can be distinguished:

- both processes are running on the same NUMA node,
- both processes are running on the same physical socket but different NUMA nodes,
- both processes are running on the same compute node but different sockets, and
- the 2 processes are running on different nodes.

You can extract more information about the system architecture by using the `system/hwloc` module.

Compilation

There are a few ways to get access to the OSU Micro-Benchmark executables. Each path of installing the benchmarks corresponds to one of the supported ways of installing in our cluster an application that links with the OpenMPI library. You should run each one of your tests with OSU Micro-Benchmark binaries source in the following manner. ￼

1. **Compiled from source** in a ReFrame genericc ompilation only test, and linking with the modules available on the ULHPC clusters.
2. **Compiled with EasyBuild** in a ReFrame EasyBuild compilation only test, and linking with the modules available on the ULHPC clusters.
3. **Binaries loaded from the EESSI distribution**, using the EESSI software stack modules [2] on the ULHPC clusters.

In each case, use the 7.2 version of OSU Micro-Benchmark suite. For the local compilation use the `foss/2023b` toolchain, available in the `env/testing/2023b` meta-module.

Project objectives

Create a set of regression tests for the bandwidth and latency in the communication between pairs of processes.

- Explore the output of `hwloc` and design a few sensible tests for the communication performance of pairs of processes running in various locations (different cores for as many different cases as you think are relevant. Focus on benchmarks running on a single node first) in the cluster.
- Implement the tests in ReFrame for all methods of sourcing a binary for the benchmarks, and both Iris (consider only IRIS regular CPU nodes) and Aion clusters.
- Run the tests and extract sensible reference performance values for each cluster, test, and binary source.

Remember, the objective of regression tests is to detect variations of the performance of the cluster. The reasons for performance variation can be installation of new driver versions for the network equipment, recompilation of the MPI libraries with different flags, and so on. Your regression tests should detect such changes.

For the purposes of our tests, we are going to focus only on a single message size per test.

- Use **message size = 8192 bytes** for the osu_latency test.
    - Typical Intra-node latency in Aion: ~2.3 μs
    - Typical Inter-node latency in Aion: ~3.9 μs
- Use **message size = 1,048,576 bytes (1MB)** for the osu_bw test.
    - Typical bandwidth in Aion: ~12,000 MB/s

Deliverables

- A set of well documented ReFrame test scripts, covering all configurations. All scripts and documentation should be contained in a Git repository and provide clear instructions on how to reproduce the results.
- A performance analysis report, including graphs, where you explain your selection of tests and performance reference values. You should explain why your reference values are expected to be stable except for the cases where variation is actually expected (e.g. update of a network library).
- Participation in a final evaluation session.

Evaluation session

The final evaluation will consist of a **30-minute session**.

- **10 minutes (max) presentation**: A concise oral presentation covering the approach, methodology, results, and key takeaways.
- **15–20 minutes**: A **live reproducibility demonstration**, where the group is expected to start from a **clean environment** (e.g., a fresh clone of the Git repository), load any required modules, and execute all relevant ReFrame tests to validate that the benchmark suite works end-to-end and results can be reproduced as documented.

Resources

1. https://mvapich.cse.ohio-state.edu/static/media/mvapich/README-OMB.txt
2. https://hpc-docs.uni.lu/software/eessi/