MVAPICH
MPI, PGAS and Hybrid MPI+PGAS Library

MVAPICH2 Meets Python: Enhancing OMB with Python Benchmarks

Presentation at MUG '21

Nawras Alnaasan

Network-Based Computing Laboratory

Dept. of Computer Science and Engineering , The Ohio State University

alnaasan.1@osu.edu

Outline

- OSU Micro Benchmarks (OMB)
- Why Python?
- MPI for Python
- Introducing OMB-Py
- Initial Results
- Conclusion

OSU Micro Benchmarks (OMB)

- OMB is a widely used package to measure performance of MPI operations.
- It helps in optimizing MPI applications on different HPC systems.
- Offers a series of benchmarks including but not limited to point-to-point, blocking and non-blocking collectives, and one-sided tests.
- A large set of options and flags for users to create customizable tests.
- Supports different platforms like ROCm/CUDA to run on ARM/NVIDIA GPUs.
- Written in C

Why Python?

- Second most popular programming language according to the TIOBE index as of August 2021.

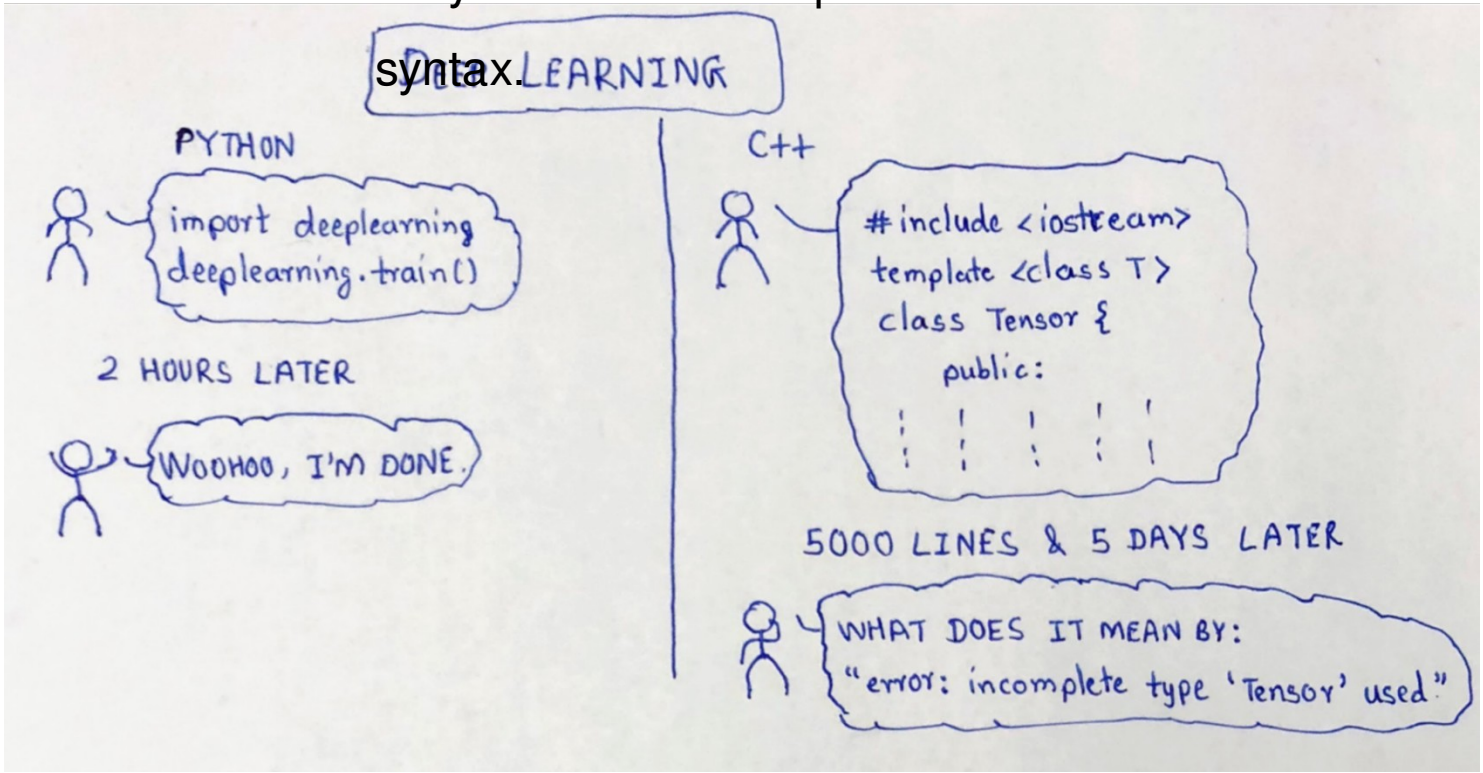
Aug 2021	Aug 2020	Change	Programming Language
1	1		C
2	3	▲	Python
3	2	▼	Java

Adopted from:
https://www.tiobe.com/tiobe-index/

- Great community and support around Machine Learning, Big Data, and Cloud Computing
  - Many deep learning frameworks like PyTorch, Tensorflow, Keras, etc.
  - One of the most important tools for data science and analytics.
- Great Potential to use MPI and distributed computing techniques.
- Many other Python libraries and frameworks like SciPy, Numpy, Django, etc.

Why Python?

Very flexible and simplified syntax.



Adopted from:
https://www.hardikp.com/2017/12/30/python-cpp/

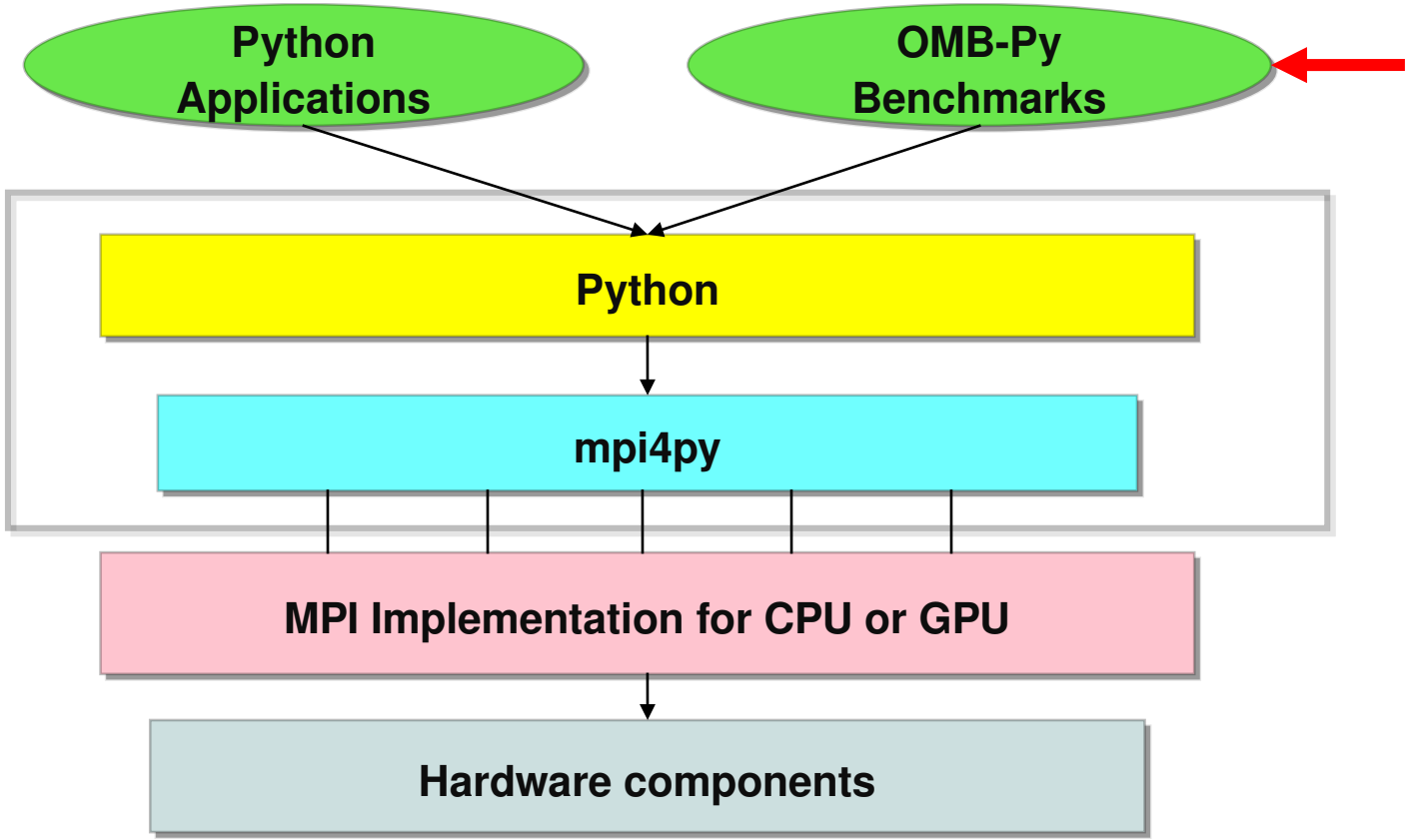
MPI for Python

- MPI support in Python is needed for multiple purposes especially for distributed computing applications (DL, data science, etc.)
- Python needs a wrapper to use MPI functionalities.
- mpi4py is the most widely used wrapper to provide Python bindings for MPI
- Other wrappers exist like the torch.distributed package from PyTorch

mpi4py

- For OMB-Py, we mainly use mpi4py to provide Python bindings for MPI.
- Supports communication for selected Python objects as memory buffers (bytearrays, Numpy, etc.)
- Capable of serializing general Python objects using built-in “Pickle” module.
  - MPI.Send() uses supported data structures as memory buffers.
  - MPI.send() serializes objects before sending data.
- Recent releases +v3.1 supports GPU-aware MPI.

Introducing OMB-Py



What does OMB-Py offer?

- Follows a similar design to the original OMB and brings it into Python.
- Support for point-to-point and collective MPI operations.
- Mimics behavior of real python applications that use MPI.
- Support for both CPU and GPU tests using CUDA-aware Python data structures as buffers like CuPy, Numba, and PyCUDA.
- A wide range of command-line arguments to run customizable tests.

Sample Latency Output

```
# OSU MPI Latency Test v5.8
# Size          Latency (us)
0                       1.12
1                       1.14
2                       1.14
4                       1.15
8                       1.14
16                      1.18
32                      1.18
64                      1.19
128                     1.25
256                     1.66
512                     1.74
1024                    1.93
2048                    2.29
4096                    3.19
8192                    4.59
16384                   6.89
32768                   9.10
65536                  12.06
131072                 21.20
262144                 32.07
524288                 54.40
1048576                97.02
2097152               181.26
4194304               351.69
```

OMB

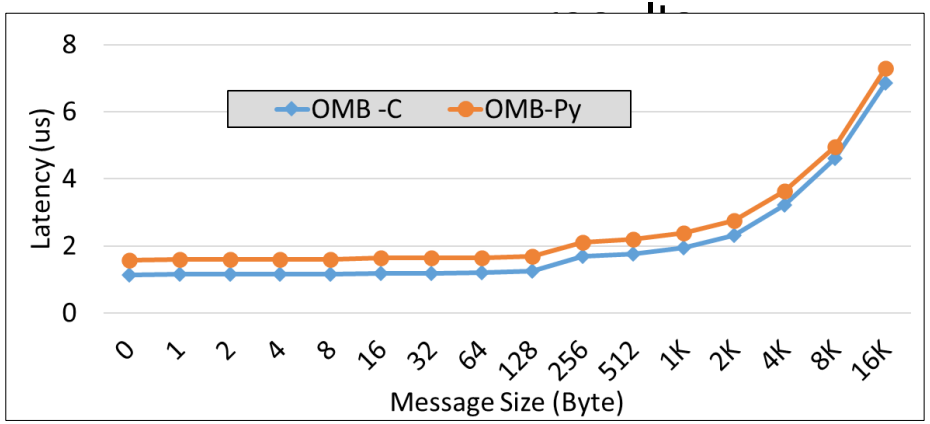
```
# OMB-Py MPI Latency Test
# Size [B]        Latency [us]
0                         1.55
1                         1.58
2                         1.58
4                         1.58
8                         1.58
16                        1.62
32                        1.62
64                        1.63
128                       1.68
256                       2.10
512                       2.18
1024                      2.36
2048                      2.73
4096                      3.63
8192                      5.03
16384                     7.24
32768                     9.79
65536                    12.73
131072                   22.17
262144                   33.01
524288                   55.19
1048576                  97.53
2097152                 186.07
4194304                 354.46
```

OMB-Py

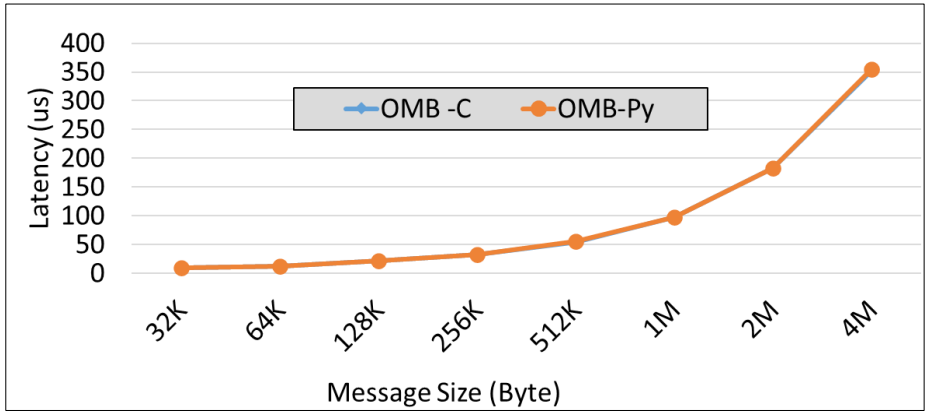
Similar CLI options

OMB-Py: Initial CPU Results

osu_latency inter-node initial results



Small message size

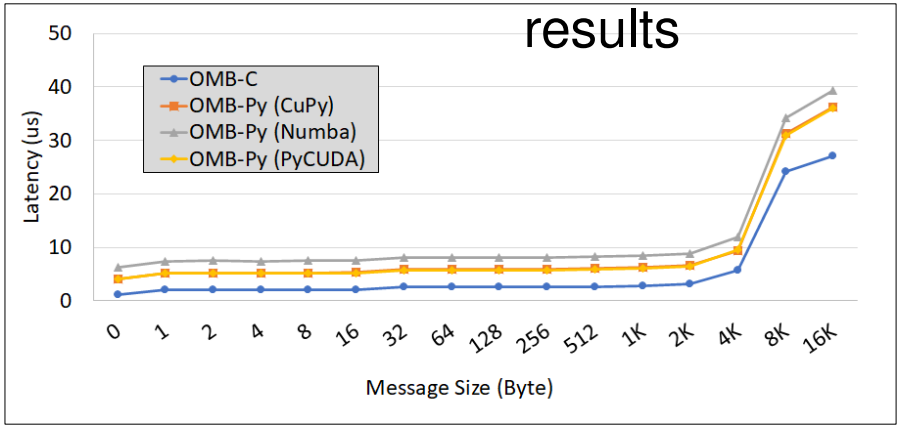


Large message size

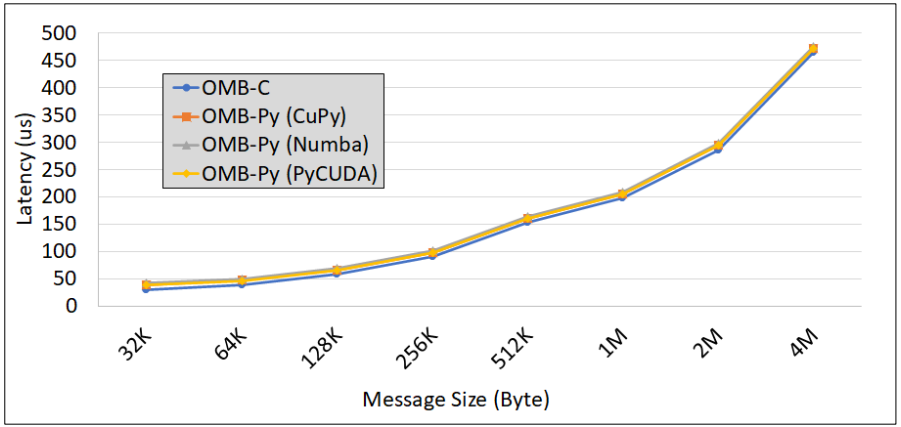
- Small Python overhead around 0.5 microseconds.
- Overhead only apparent in smaller message sizes

OMB-Py: Initial GPU Results

osu_latency inter-node initial results



Small message size



Large message size

- Different overheads depending on the CUDA-aware data structure
  - Around 3 microseconds for CuPy and PyCUDA
  - Around 5 microseconds for Numba
- Overhead is only apparent in smaller message sizes

Conclusion

- MPI support for Python is crucially needed in the Python community.
- MPI benchmarks are needed to measure and optimize MPI performance.
- mpi4py provides Python bindings for MPI and is used with OMB-Py.
- OMB-Py provides Python MPI benchmarking similar to OMB in C.
- Initial results show a small overhead due to Python-MPI bindings layer and the use of Python objects as buffers for communication.

Thank You!

alnaasan.1@osu.edu

Network-Based Computing Laboratory
http://nowlab.cse.ohio-state.edu/

High Performance Deep Learning
http://hidl.cse.ohio-state.edu/

The High-Performance Deep Learning Project
http://hidl.cse.ohio-state.edu/

The High-Performance MPI/PGAS Project
http://mvapich.cse.ohio-state.edu/