MPI for Python and the new MPI ABI

جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

Lisandro Dalcin
dalcinl@gmail.com

EuroMPI/Australia 2024

What is MPI for Python (**mpi4py**)?

- Unofficial Python bindings for MPI
- API based on MPI-2 C++ bindings (didn’t age well)
- Almost all MPI features are supported
  - Best suited for MPI-4 implementations
  - Also works with MPI-1, MPI-2, MPI-3 implementations

Features - MPI-1

- Point to point communication
  - blocking (send/recv)
  - nonblocking (isend/irecv + test/wait)
- Collective operations
  - Synchronization (barrier)
  - Communication (broadcast, scatter/gather)
  - Global reductions (reduce, scan)
- Process groups, communication domains, virtual topologies

Features - MPI-2

- Parallel I/O (read/write)
- Dynamic process management (spawn, connect/accept)
- One-sided operations, a.k.a. RMA (put/get/accumulate)

Features - MPI-3

- Matching probes (thread-safety)
- Non blocking collectives (ibarrier, ibcast, igatter)
- Neighbor collectives (neighbor allgather/alltoall)

Features - MPI-4

- Persistent collectives
- Partitioned communication
- Sessions
- Large-count APIs, finally! 🚀

Current MPI API (excluding MPI_T)

	constants & routines
MPI-1	242
MPI-2	357
MPI-3	108
MPI-4	91 162
	960

Hello World!

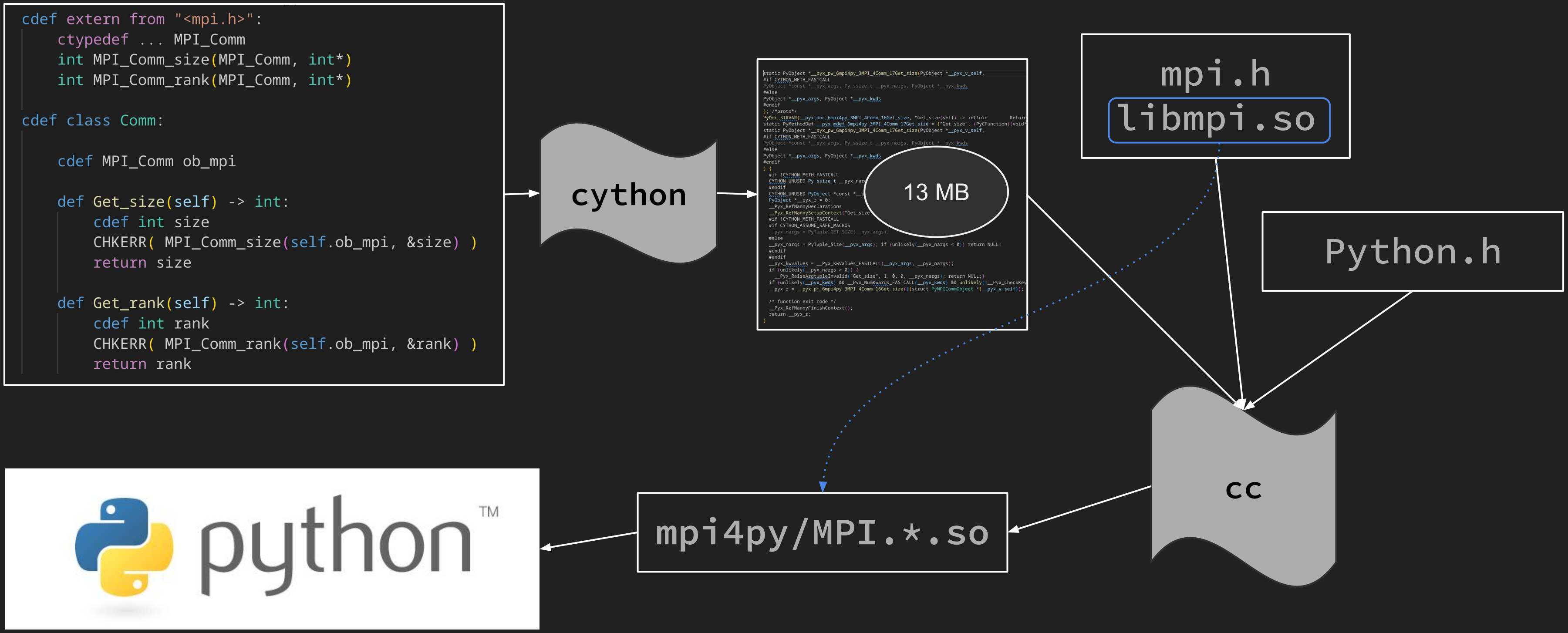
```python
from mpi4py import MPI

rank = MPI.COMM_WORLD.Get_rank()
size = MPI.COMM_WORLD.Get_size()
name = MPI.Get_processor_name()

print(
    "Hello, World! I am process",
    f"{rank} of {size} on {name}"
)
```

```
$ mpiexec -n 5 python helloworld.py
```

From source to binary - `pip install mpi4py`



```
cdef extern from "<mpi.h>":
    ctypedef ... MPI_Comm
    int MPI_Comm_size(MPI_Comm, int*)
    int MPI_Comm_rank(MPI_Comm, int*)

cdef class Comm:

    cdef MPI_Comm ob_mpi

    def Get_size(self) -> int:
        cdef int size
        CHKERR( MPI_Comm_size(self.ob_mpi, &size) )
        return size

    def Get_rank(self) -> int:
        cdef int rank
        CHKERR( MPI_Comm_rank(self.ob_mpi, &rank) )
        return rank
```

```
static PyObject *__pyx_pw_6mpi4py_3MPI_4Comm_17Get_size(PyObject *__pyx_v_self,
#if CYTHON_METH_FASTCALL
PyObject *const *__pyx_args, Py_ssize_t __pyx_nargs, PyObject *__pyx_kwds
#else
PyObject *__pyx_args, PyObject *__pyx_kwds
#endif
); /*proto*/
PyDoc_STRVAR(__pyx_doc_6mpi4py_3MPI_4Comm_16Get_size, "Get_size(self) -> int\n\n        Return
static PyMethodDef __pyx_mdef_6mpi4py_3MPI_4Comm_17Get_size = {"Get_size", (PyCFunction)(void*
static PyObject *__pyx_pw_6mpi4py_3MPI_4Comm_17Get_size(PyObject *__pyx_v_self,
#if CYTHON_METH_FASTCALL
PyObject *const *__pyx_args, Py_ssize_t __pyx_nargs, PyObject *__pyx_kwds
#else
PyObject *__pyx_args, PyObject *__pyx_kwds
#endif
) {
  #if !CYTHON_METH_FASTCALL
  CYTHON_UNUSED Py_ssize_t __pyx_narg
  #endif
  CYTHON_UNUSED PyObject *const *__p
  PyObject *__pyx_r = 0;
  __Pyx_RefNannyDeclarations
  __Pyx_RefNannySetupContext("Get_size
  #if !CYTHON_METH_FASTCALL
  #if CYTHON_ASSUME_SAFE_MACROS
  __pyx_nargs = PyTuple_GET_SIZE(__pyx_args);
  #else
  __pyx_nargs = PyTuple_Size(__pyx_args); if (unlikely(__pyx_nargs < 0)) return NULL;
  #endif
  #endif
  __pyx_kwvalues = __Pyx_KwValues_FASTCALL(__pyx_args, __pyx_nargs);
  if (unlikely(__pyx_nargs > 0)) {
    __Pyx_RaiseArgtupleInvalid("Get_size", 1, 0, 0, __pyx_nargs); return NULL;}
  if (unlikely(__pyx_kwds) && __Pyx_NumKwargs_FASTCALL(__pyx_kwds) && unlikely(!__Pyx_CheckKey
  __pyx_r = __pyx_pf_6mpi4py_3MPI_4Comm_16Get_size(((struct PyMPICommObject *)__pyx_v_self));

  /* function exit code */
  __Pyx_RefNannyFinishContext();
  return __pyx_r;
}
```

Third-party pre-built mpi4py binaries

- macOS: Homebrew (Open MPI) and MacPorts (MPICH)
- Linux distributions: MPICH or Open MPI or both (via modules/alternatives)
- conda-forge https://github.com/conda-forge/mpi4py-feedstock

mpi4py-feedstock/recipe/conda_build_config.yaml

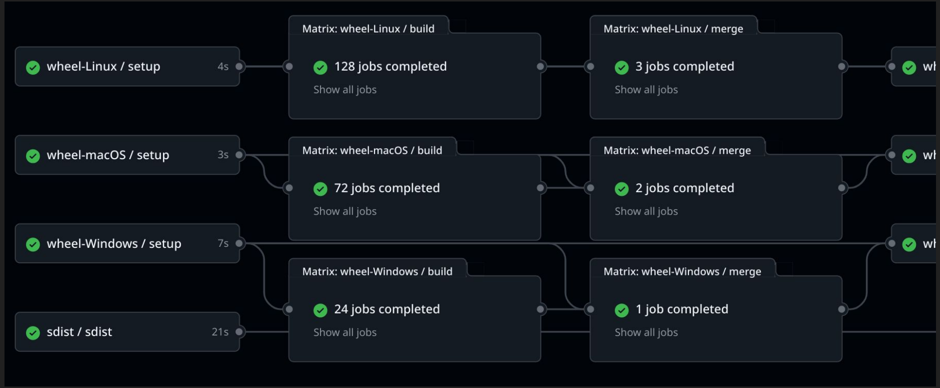
```
mpi:
  - mpich     # [not win]
  - openmpi   # [not win]
  - impi      # [not osx and x86_64]
  - msmpi     # [win]
```

65 builds

Building binary Python wheels for distribution

https://github.com/mpi4py/mpi4py-publish

Python 3.6 - 3.13 PyPy 3.7 - 3.10 MPI 3.1 / 4.0 / 4.1		
Linux	x86_64 aarch64 ppc64le	MPICH Open MPI (Intel MPI)
macOS	arm64 x86_64	MPICH Open MPI
Windows	AMD64	Intel MPI MSMPI

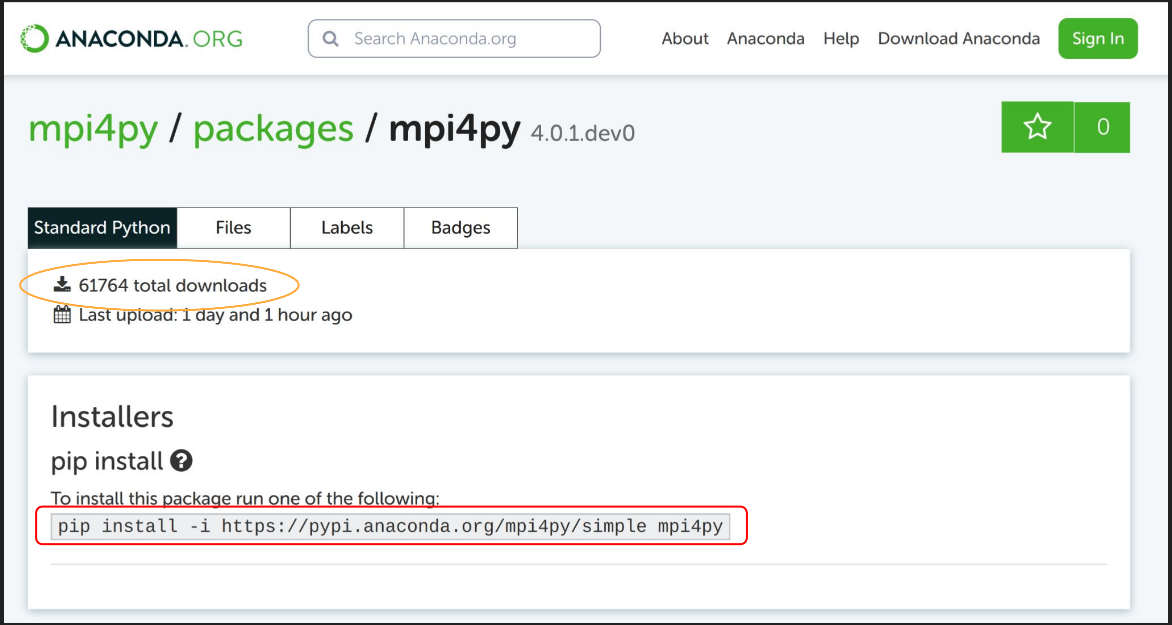


```
mpi4py-4.0.0-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
```

```
mpi4py/__init__.py
. . .
mpi4py/MPI.mpi31-mpich.cpython-312-x86_64-linux-gnu.so
mpi4py/MPI.mpi40-mpich.cpython-312-x86_64-linux-gnu.so
mpi4py/MPI.mpi41-mpich.cpython-312-x86_64-linux-gnu.so
mpi4py/MPI.mpi31-openmpi.cpython-312-x86_64-linux-gnu.so
```

Distributing binary Python wheels

`https://anaconda.org/mpi4py/mpi4py`



mpi4py / packages / mpi4py 4.0.1.dev0

61764 total downloads

Last upload: 1 day and 1 hour ago

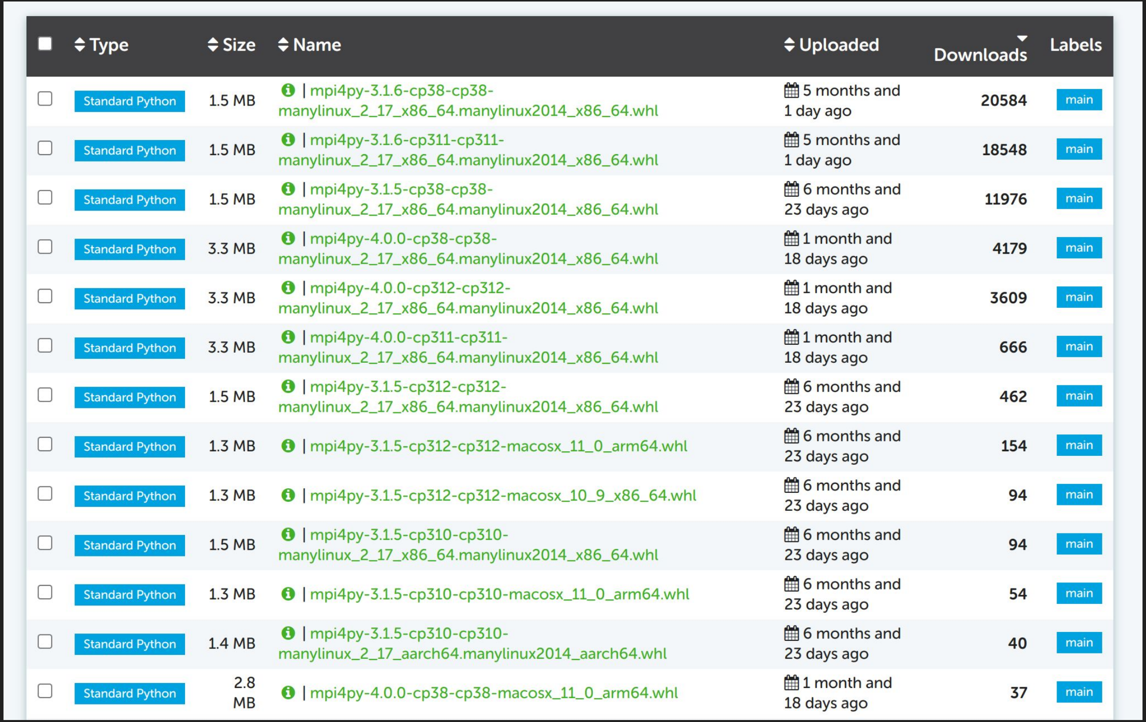
Installers

pip install

To install this package run one of the following:

```
pip install -i https://pypi.anaconda.org/mpi4py/simple mpi4py
```

Is this useful? Does anyone cares?



	Type	Size	Name	Uploaded	Downloads	Labels
☐	Standard Python	1.5 MB	mpi4py-3.1.6-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl	5 months and 1 day ago	20584	main
☐	Standard Python	1.5 MB	mpi4py-3.1.6-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl	5 months and 1 day ago	18548	main
☐	Standard Python	1.5 MB	mpi4py-3.1.5-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl	6 months and 23 days ago	11976	main
☐	Standard Python	3.3 MB	mpi4py-4.0.0-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl	1 month and 18 days ago	4179	main
☐	Standard Python	3.3 MB	mpi4py-4.0.0-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl	1 month and 18 days ago	3609	main
☐	Standard Python	3.3 MB	mpi4py-4.0.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl	1 month and 18 days ago	666	main
☐	Standard Python	1.5 MB	mpi4py-3.1.5-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl	6 months and 23 days ago	462	main
☐	Standard Python	1.3 MB	mpi4py-3.1.5-cp312-cp312-macosx_11_0_arm64.whl	6 months and 23 days ago	154	main
☐	Standard Python	1.3 MB	mpi4py-3.1.5-cp312-cp312-macosx_10_9_x86_64.whl	6 months and 23 days ago	94	main
☐	Standard Python	1.5 MB	mpi4py-3.1.5-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl	6 months and 23 days ago	94	main
☐	Standard Python	1.3 MB	mpi4py-3.1.5-cp310-cp310-macosx_11_0_arm64.whl	6 months and 23 days ago	54	main
☐	Standard Python	1.4 MB	mpi4py-3.1.5-cp310-cp310-manylinux_2_17_aarch64.manylinux2014_aarch64.whl	6 months and 23 days ago	40	main
☐	Standard Python	2.8 MB	mpi4py-4.0.0-cp38-cp38-macosx_11_0_arm64.whl	1 month and 18 days ago	37	main

Preliminary MPI ABI support

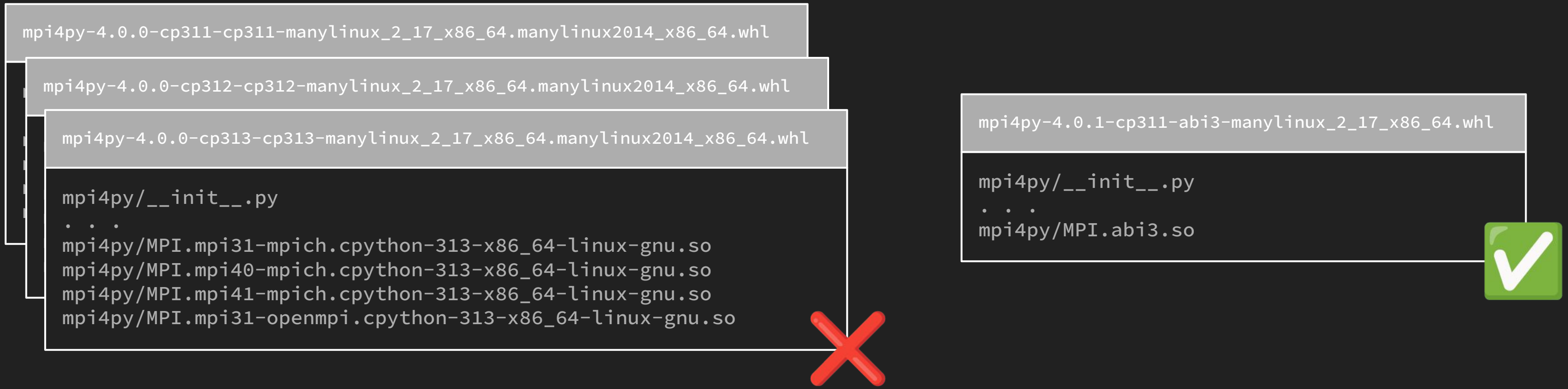
https://github.com/mpi4py/mpi4py-testing/actions/workflows/abi.yml

- Build step
  - Use MPI stubs (https://github.com/mpiwg-abi/header_and_stub_library)
  - Generate Python wheel the usual way
- Test step
  - Build MPICH with `./configure --enable-mpi-abi`
  - Install Python wheel from build step
  - Run full mpi4py test suite

A Future With (MPI ABI) Hope

- Use MPI API/ABI with weak symbols (multiple MPI std versions)
- Use `Py_LIMITED_API` (mpi4py would require Python >= 3.11)

Just one Python wheel per OS & Arch!



```
mpi4py-4.0.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
mpi4py-4.0.0-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
mpi4py-4.0.0-cp313-cp313-manylinux_2_17_x86_64.manylinux2014_x86_64.whl

mpi4py/__init__.py
. . .
mpi4py/MPI.mpi31-mpich.cpython-313-x86_64-linux-gnu.so
mpi4py/MPI.mpi40-mpich.cpython-313-x86_64-linux-gnu.so
mpi4py/MPI.mpi41-mpich.cpython-313-x86_64-linux-gnu.so
mpi4py/MPI.mpi31-openmpi.cpython-313-x86_64-linux-gnu.so
```

❌

```
mpi4py-4.0.1-cp311-abi3-manylinux_2_17_x86_64.whl

mpi4py/__init__.py
. . .
mpi4py/MPI.abi3.so
```

✅

Time Machine

If I could go back in time carrying the MPI ABI in my pocket…

- Get rid of dealing with C compilers and pre-built binaries
- Dynamically load the MPI library at runtime
- Call MPI routines via `ctypes` (built-in) or `cffi` (much nicer)
- Pure Python code working with any MPI ABI implementation
- Slightly slower in hotspots (usual Python overhead)

Thanks!

https://github.com/mpi4py/mpi4py

https://anaconda.org/mpi4py/mpi4py

https://github.com/mpiwg-abi/abi-issues

https://github.com/mpiwg-abi/header_and_stub_library