# Using Two-Sided Blocking Point to Point Communications in MPI

**Erik Saule**
esaule@uncc.edu

Parallel and Distributed Computing

# Learning Outcomes

**After this lecture you will be able to**

- Use blocking point to point communication primitives.

# Outline


1. Idea
2. Syntax
3. Underspecifying MPI_Recv
4. Further

# Two-sided communications

## What?

- Make a communication between a pair of processes
- The sender makes a call to send
- The received makes a different call to receive
- This breaks process symmetry :(

## Why?

- Most engineering simulation
- Manager Worker Systems like Folding at Home
- Any application structure with dedicated nodes (database, IO node, specialized units like GPUs)

# Outline


1. Idea
2. Syntax
3. Underspecifying MPI_Recv
4. Further

# MPI_Send

man MPI_Send

# MPI_Send

man MPI_Recv

# Simple example

```
#include <mpi.h>
#include <iostream>
#include <vector>

void printvector(const std::string & msg, const std::vector<int>& v) {
  std::cout<<msg;
  for (auto& a : v)
    std::cout<<a<<" ";
  std::cout<<"\n";
}

int main (int argc, char*argv[])  {
  MPI_Init (&argc, &argv);
  int size;
  int rank;
  MPI_Comm_size(MPI_COMM_WORLD, &size);
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);

  std::cout<<"I am "<<rank<<" out of "<<size<<"\n";

  int vecsize = 3;

  std::vector<int> vec(vecsize);
  std::vector<int> vec2(vecsize);

  for (int i=0; i<vecsize; ++i)
    vec[i] = rank*100+i;

  printvector("before: ", vec);

  if (rank == 0) {
    MPI_Send(&(vec[0]), vecsize, MPI_INT, 1, 123, MPI_COMM_WORLD);
```

# Outline

1 Idea

2 Syntax

3 Underspecifying MPI_Recv

4 Further

# What's up with MPI_Status?

## Because recv takes wildcards

- Source can be MPI_ANY_SOURCE
- Tag can be MPI_ANY_TAG
- Count just needs to be larger than what is being sent

## How to know the unknown?

- status.MPI_SOURCE
- status.MPI_TAG
- int MPI_Get_count(const MPI_Status *status, MPI_Datatype datatype, int *count)

# Usage example of status

```
#include <mpi.h>
#include <iostream>
#include <vector>

void printvector(const std::string & msg, const std::vector<int>& v) {
  std::cout<<msg;
  for (auto& a : v)
    std::cout<<a<<" ";
  std::cout<<"\n";
}

int main (int argc, char*argv[])  {
  MPI_Init (&argc, &argv);

  int size;
  int rank;

  MPI_Comm_size(MPI_COMM_WORLD, &size);
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);

  std::cout<<"I am "<<rank<<" out of "<<size<<"\n";

  int vecsize = 3;

  std::vector<int> vec(vecsize);
  std::vector<int> vec2(vecsize*10);

  for (int i=0; i<vecsize; ++i)
    vec[i] = rank*100+i;

  printvector("before: ", vec);
```

# Outline


1. Idea
2. Syntax
3. Underspecifying MPI_Recv
4. Further

# External

Books:

- Using MPI, 3rd edition. William Gropp, Ewing Lusk and Anthony Skjellum. MIT Press. Available through the library at https://librarylink.uncc.edu/login?url=http://ieeexplore.ieee.org/xpl/bkabstractplus.jsp?bkn=6981847
- Using Advanced MPI. William Gropp, Torsten Hoefler, Rajeev Thakur and Ewing Lusk. MIT Press. Available through the library at https://librarylink.uncc.edu/login?url=http://ieeexplore.ieee.org/xpl/bkabstractplus.jsp?bkn=6981848

MPI implementations:

- MPICH https://www.mpich.org
- OpenMPI https://www.open-mpi.org/

API Documentation:

- MPICH man pages https://www.mpich.org/static/docs/v3.2/www3/index.htm
- OpenMPI documentation https://www.open-mpi.org/doc/v3.0/

Slides from colleagues:

- Tutorial on MPI programming. Victor Eijkhout. https://bitbucket.org/VictorEijkhout/parallel-computing-book/raw/e11748c8d8ae874ed645566ba0e82aa787ecf959/EijkhoutMPIlecture.pdf
- MPI for Dummies. Pavan Balaki, Torsten Hoefler. https://htor.inf.ethz.ch/teaching/mpi_tutorials/ppopp13/2013-02-24-ppopp-mpi-basic.pdf

Tutorial:

- part one of Parallel Programming in MPI and OpenMP. Victor Eijkhout. Draft at https://bitbucket.org/VictorEijkhout/parallel-computing-book/raw/e11748c8d8ae874ed645566ba0e82aa787ecf959/EijkhoutParComp.pdf