MPI Quick Reference Guide – JMU CS 470

General

```
int MPI_Init (int *argc, char ***argv)
int MPI_Finalize ()
int MPI_Barrier (MPI_Comm comm)
double MPI_Wtime ()
```

```
int MPI_Comm_size (MPI_Comm comm, int *size)
int MPI_Comm_rank (MPI_Comm comm, int *rank)
```

Default communicator: **MPI_COMM_WORLD**

```
struct MPI_STATUS {
  int MPI_SOURCE
  int MPI_TAG
  int MPI_ERROR
}
```

Point-to-point Operations

```
int MPI_Send  (void *buf, int count, MPI_Datatype dtype, int dest, int tag, MPI_Comm comm)
int MPI_Ssend (void *buf, int count, MPI_Datatype dtype, int dest, int tag, MPI_Comm comm)
int MPI_Recv  (void *buf, int count, MPI_Datatype dtype, int src,  int tag, MPI_Comm comm, MPI_Status *status)
                          (maximum count)                (MPI_ANY_SOURCE / MPI_ANY_TAG)              (MPI_STATUS_IGNORE)

int MPI_Sendrecv (void *send_buf, int send_count, MPI_Datatype send_dtype, int dest, int send_tag
                  void *recv_buf, int recv_count, MPI_Datatype recv_dtype, int src,  int recv_tag,
                  MPI_Comm comm, MPI_Status *status)

int MPI_Isend (void *buf, int count, MPI_Datatype dtype, int dest, int tag, MPI_Comm comm, MPI_Request *request)
int MPI_Irecv (void *buf, int count, MPI_Datatype dtype, int src,  int tag, MPI_Comm comm, MPI_Request *request,
               MPI_Status *status)

int MPI_Test (MPI_Request *request, int *flag, MPI_Status *status)
int MPI_Wait (MPI_Request *request,            MPI_Status *status)
int MPI_Get_count (MPI_Status *status, MPI_Datatype dtype, int *count)
```

Collective Operations

```
int MPI_Bcast     (void      *buf,                int count, MPI_Datatype dtype,                int root, MPI_Comm comm)

int MPI_Reduce    (void *send_buf, void *recv_buf, int count, MPI_Datatype dtype, MPI_Op op, int root, MPI_Comm comm)
int MPI_Allreduce (void *send_buf, void *recv_buf, int count, MPI_Datatype dtype, MPI_Op op,           MPI_Comm comm)

int MPI_Scatter   (void *send_buf,            int send_count, MPI_Datatype send_dtype,
                   void *recv_buf,            int recv_count, MPI_Datatype recv_dtype,       int root, MPI_Comm comm)

int MPI_Gather    (void *send_buf,            int send_count, MPI_Datatype send_dtype,
                   void *recv_buf,            int recv_count, MPI_Datatype recv_dtype,       int root, MPI_Comm comm)

int MPI_Allgather (void *send_buf,            int send_count, MPI_Datatype send_dtype,
                   void *recv_buf,            int recv_count, MPI_Datatype recv_dtype,                 MPI_Comm comm)

int MPI_Alltoall  (void *send_buf,            int send_count, MPI_Datatype send_dtype,
                   void *recv_buf,            int recv_count, MPI_Datatype recv_dtype,                 MPI_Comm comm)
```

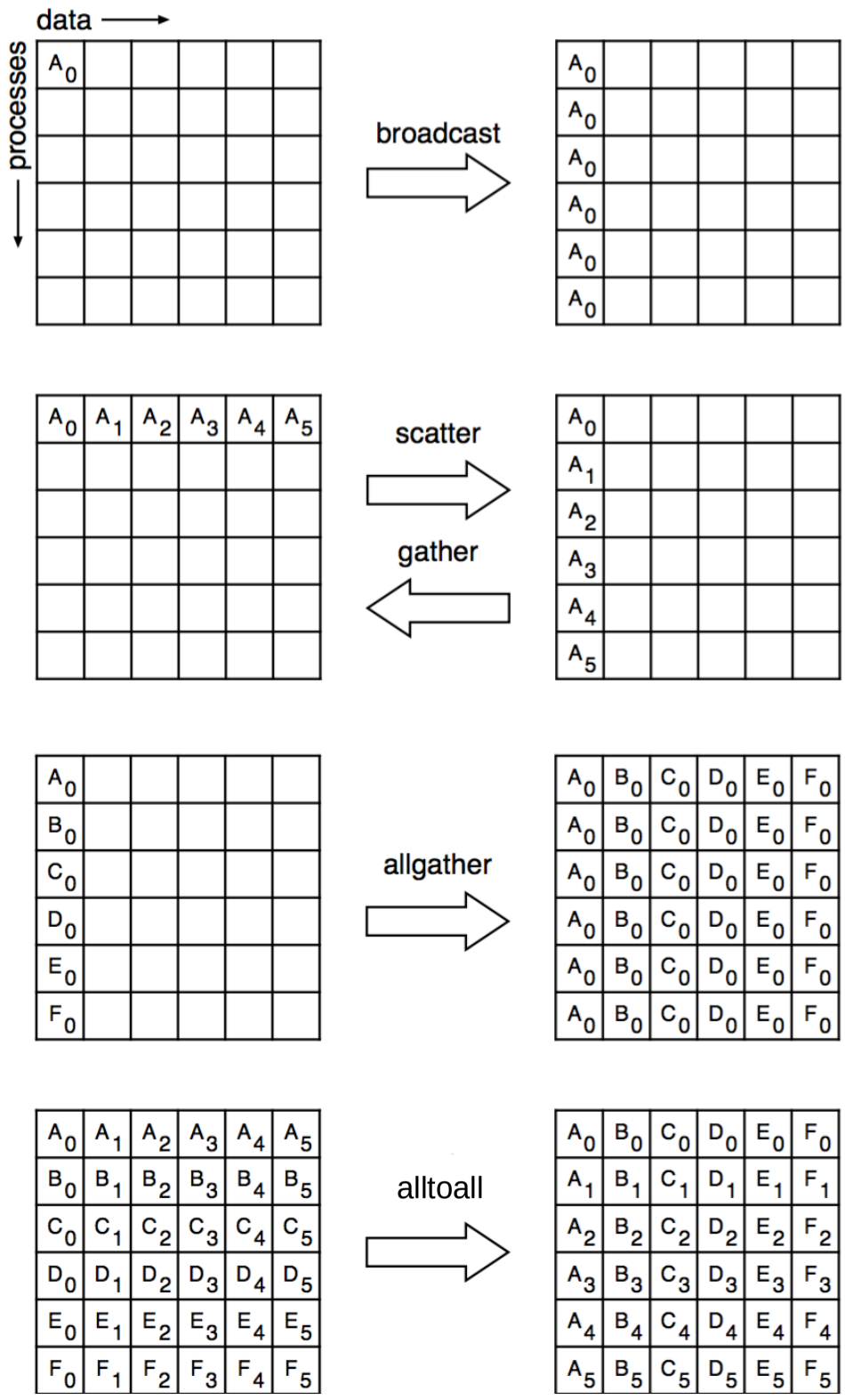

Figure 5.1: Collective move functions illustrated for a group of six processes. In each case, each row of boxes represents data locations in one process. Thus, in the broadcast, initially just the first process contains the data A_0, but after the broadcast all processes contain it.

Figure from MPI-3.1 standard (June 4, 2015):
`http://www.mpi-forum.org/docs/mpi-3.1/mpi31-report.pdf`