Virtual Workshop

MPI Collective Communications: Scatterv Syntax

We next discuss the precise MPI syntax of a couple of “v” calls. It will be useful to know the entire calling sequence when completing the exercise at the end of this module.

MPI_Scatterv syntax

The MPI_Scatterv calling sequence is really pretty straightforward—perhaps (dare we say) self-explanatory? Anyway, here’s what it looks like in Fortran.

```
INTEGER SENDCOUNTS(0:NPROC-1), DISPLS(0:NPROC-1)
...
CALL MPI_SCATTERV( &
       SENDBUF, SENDCOUNTS, DISPLS, SENDTYPE, &
       RECVBUF, RECVCOUNT, RECVTYPE, &
       ROOT, COMM, IERROR )
```

Compared to just plain scatter, there are two differences: the SENDCOUNTS argument has become an array, and a new argument DISPLS has been added to the list.

> SENDCOUNTS(I) is the number of items of type SENDTYPE
> to send from process ROOT to process I.
> Thus its value is significant only on ROOT.
> DISPLS(I) is the displacement from SENDBUF
> to the beginning of the I-th message, in units of SENDTYPE.
> It also has significance only for the ROOT process.

And that's pretty much it! The other arguments in the list simply mirror the usual MPI_Scatter calling sequence. Here is the C syntax, which differs only in minor C-like details:

```
  int sendcounts[NPROC], displs[NPROC];
  ...
  MPI_Scatterv(
    sendbuf, sendcounts, displs, sendtype,
    recvbuf, recvcount, recvtype,
    root, comm);
```

MPI_Gatherv syntax

Let's turn now to gather and gatherv, which are the exact inverses of scatter and scatterv. In fact, we don't even need new diagrams! Simply think about reversing the directions of the arrows in the previous diagrams. We will therefore turn our attention directly to the syntax of the gatherv call. Again, we'll look at it in Fortran first:

```
INTEGER RECVCOUNTS(0:NPROC-1), DISPLS(0:NPROC-1)
...
CALL MPI_GATHERV( &
       SENDBUF, SENDCOUNT, SENDTYPE, &
       RECVBUF, RECVCOUNTS, DISPLS, RECVTYPE, &
       ROOT, COMM, IERROR )
```

Here, RECVCOUNTS(I) plays the role that SENDCOUNTS(I) played in the call to MPI_Scatterv. Its location in the argument list has been shifted accordingly, to put it among the arguments relating to the receiver. DISPLS(I) in this case indicates where to place the data arriving from process I. It is given as an offset relative to address RECVBUF on the ROOT process, and it is in units of SENDTYPE. Therefore, compared to MPI_Gather, there is an additional array DISPLS, while RECVCOUNTS has been stretched into an array.

Here is what it looks like in C:

```
  int recvcounts[NPROC], displs[NPROC];
  ...
  MPI_Scatterv(
    sendbuf, sendcount, sendtype,
    recvbuf, recvcounts, displs, recvtype,
    root, comm);
```

<= previous next =>

Add my notes

Mark (M) my place in this topic