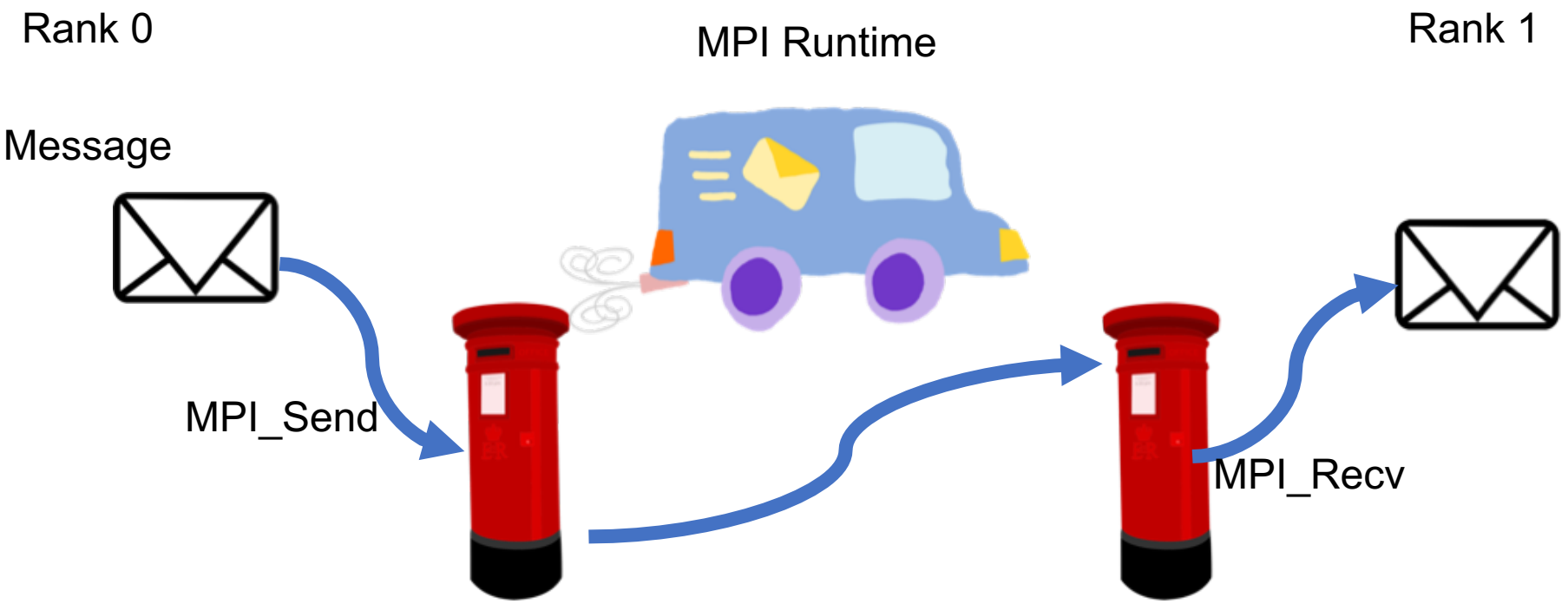
Message Passing



6 Basic Functions

```
MPI_Init ()

MPI_Comm_rank (Comm)

MPI_Comm_size (Comm)

MPI_Send (Data, Num, Type, Dest, Tag, Comm)

MPI_Recv (Data, Num, Type, Src, Tag, Comm, Stat)

MPI_Finalize ()
```

Communication type

Point-to-Point communication

Collective communication

Blocking communication

Non-blocking communication

Non-blocking communication

```
MPI_Isend (Data, Num, Type, Dest, Tag, Comm, &Req)

MPI_Irecv (Data, Num, Type, Src, Tag, Comm, &Req)

MPI_Wait (&Req, &Status)

MPI_Waitall (Num, Req[], Stat[])
```

Collective Communication

```
MPI_Bcast
MPI_Ibcast

MPI_Scatter
MPI_Gather

MPI_Allscatter
MPI_Alltoall
```

Collective Communication

```
rank    send buf                              recv buf
----    --------                              --------
 0      a,b,c          MPI_Allgather          a,b,c,A,B,C,#,@,%
 1      A,B,C         ----------------->      a,b,c,A,B,C,#,@,%
 2      #,@,%                                 a,b,c,A,B,C,#,@,%
```

```
rank    send buf                              recv buf
----    --------                              --------
 0      a,b,c           MPI_Alltoall          a,A,#
 1      A,B,C          ----------------->     b,B,@
 2      #,@,%                                 c,C,%

(a more elaborate case with two elements per process)

rank    send buf                              recv buf
----    --------                              --------
 0      a,b,c,d,e,f     MPI_Alltoall          a,b,A,B,#,@
 1      A,B,C,D,E,F    ----------------->     c,d,C,D,%,$
 2      #,@,%,$,&,*                           e,f,E,F,&,*
```

Example

Rank i wants to send message i to all other ranks

Implementation 1 : Blocking one-to-one:

```
for (int irank = 0; irank < size; ++irank) {
  if (irank == rank) continue;
  int stag = irank + size * rank;
  MPI_Send (&sendbuff, 1, MPI_INT, irank, stag, MPI_COMM_WORLD);
  std::cout << "rank " << rank << " sending message " << sendbuff << " to rank " << irank << std::endl;
  int rtag = rank + size * irank;
  MPI_Recv (&(recvbuff[irank]), 1, MPI_INT, irank, rtag, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
}
```

Notice encoding of sender and receiver in tag!

DANGER : Deadlock!

Example

Rank i wants to send message i to all other ranks

Implementation 2 : Non-Blocking one-to-one:

```
for (int irank = 0; irank < size; ++irank) {
  if (irank == rank) continue;
  int stag = irank + size * rank;
  MPI_Isend (&sendbuff, 1, MPI_INT, irank, stag, MPI_COMM_WORLD, &(reqs[numreq++]));
  std::cout << "rank " << rank << " sending message " << sendbuff << " to rank " << irank << std::endl;
  int rtag = rank + size * irank;
  MPI_Irecv (&(recvbuff[irank]), 1, MPI_INT, irank, rtag, MPI_COMM_WORLD, &(reqs[numreq++]));
}
MPI_Waitall (numreq, &(reqs[0]), MPI_STATUSES_IGNORE);
```

Notice encoding of sender and receiver in tag!

Notice Use of Waitall!

Example

Rank i wants to send message i to all other ranks

Implementation 3 : Blocking Collective (All-to-all)

```
recvbuff.assign (recvbuff.size (), rank);
MPI_Alltoall (MPI_IN_PLACE, 1, MPI_INT, &(recvbuff[0]), 1, MPI_INT, MPI_COMM_WORLD);
```

Notice MPI_IN_PLACE

Very convenient when messages of same known size are to be sent / received!

Example

Rank i wants to send message i to all other ranks

Implementation 4 : Blocking Collective (All-gather)

```
recvbuff.assign (recvbuff.size (), rank);
MPI_Allgather (&sendbuff, 1, MPI_INT, &(recvbuff[0]), 1, MPI_INT, MPI_COMM_WORLD);
```

Very convenient when messages of same known size are to be sent / received!

Advanced Example

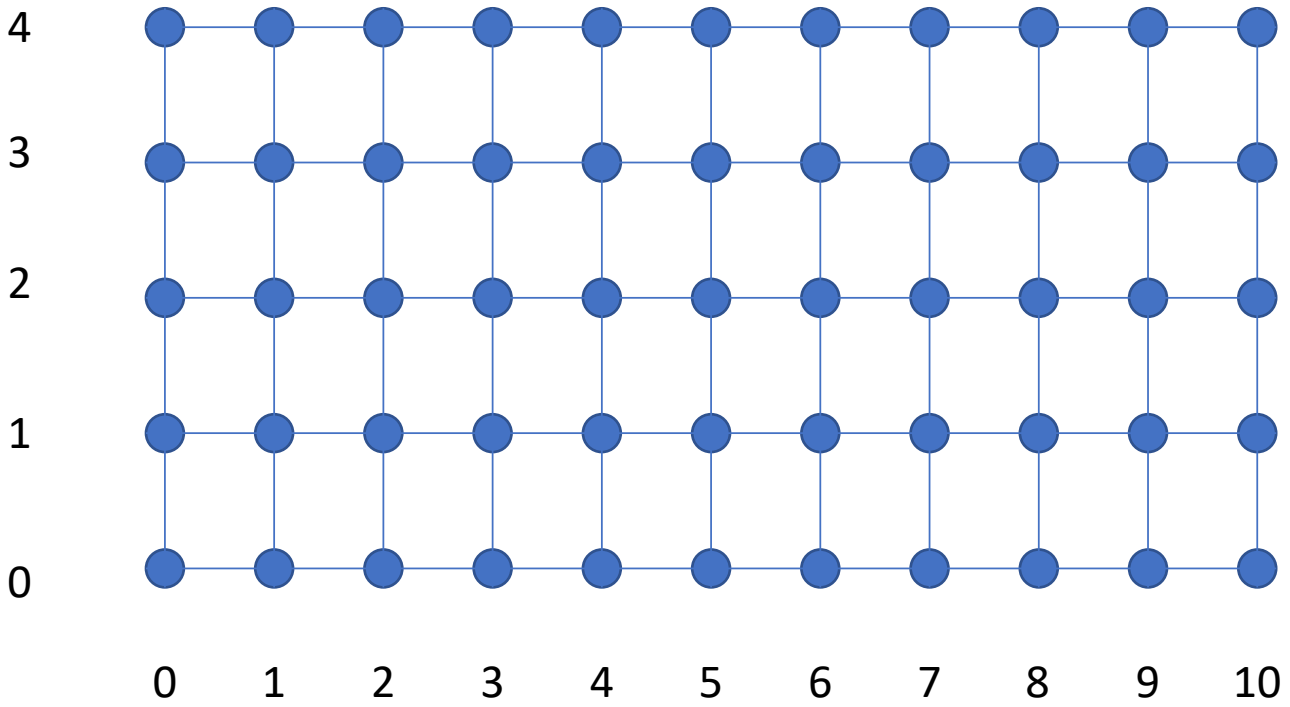
Distributed assembly of sparse_matrix :

- Each rank **owns** a contiguous range of rows
- Each rank can **assign** elements within its ownership range or **outside** its ownership range
- If multiple ranks assign one matrix element, after assembly the actual value must be the **sum** of local contributions

This is the most ommon situation encountered in **Finite Element** problems

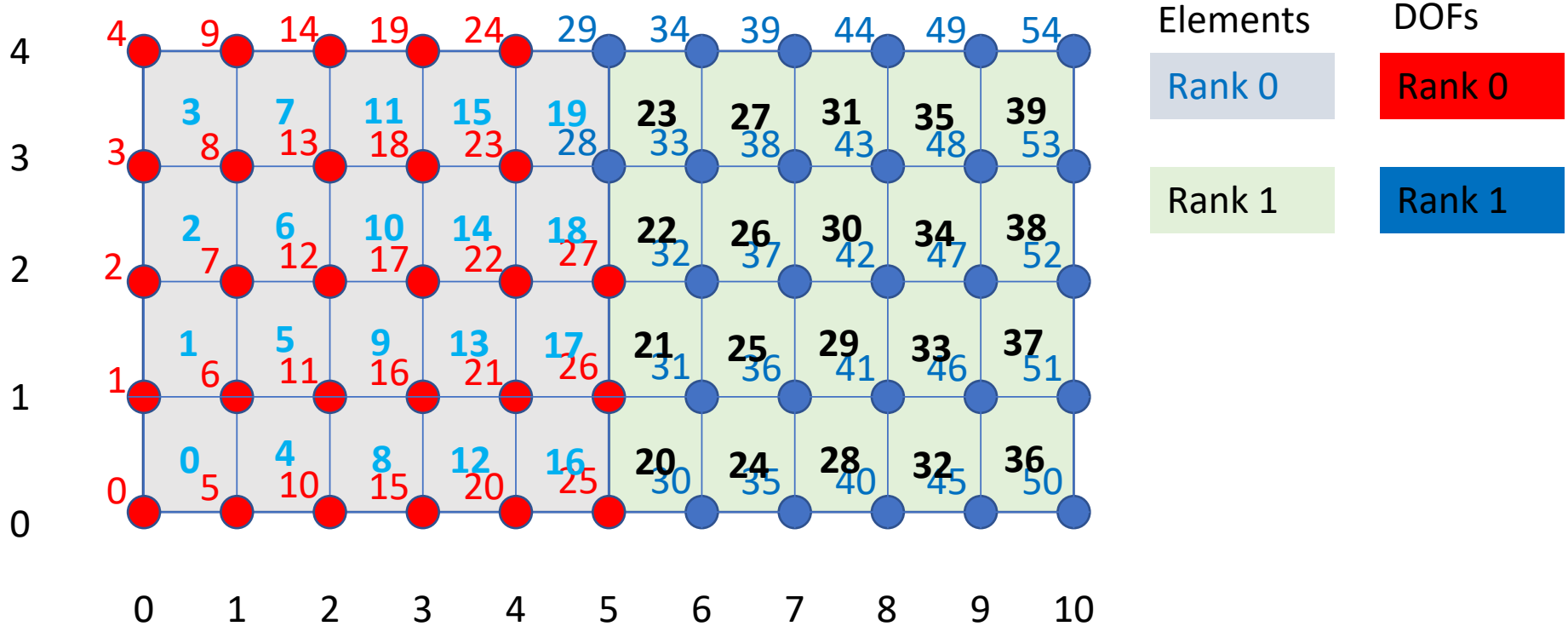
- Each rank **owns** a range of mesh **elements** and a range of **DOFs**
- For low order continuous formulations **DOFs** correspond to mesh **vertices**
- The same vertex touches multiple **elements**
- The matrix entries related to a given DOF are the sum of contributions from all elements touching the corresponding vertex

Exercise : assemble FEM Laplacian



	J0	J1	J2	J3
I0	2	-1	-1	
I1	-1	2		-1
I2	-1		2	-1
I3		-1	-1	2

Exercise : assemble FEM Laplacian



	J0	J1	J2	J3
I0	2	-1	-1	
I1	-1	2		-1
I2	-1		2	-1
I3		-1	-1	2