Recently covered

- Blocking point-to-point routines (send, ssend, recv, sendrecv, sendrecv_replace)
- Collective routines (barrier, bcast, gather/scatter, alltoall, reduce, allreduce)
- Non-blocking point-to-point routines (isend, irecv, issend)
- Persistent comm (special non-blocking)

Sending non-contiguous data

The routines `mpi_pack, mpi_unpack` are available but not recommended. MPI derived types provide an elegant way to send/recv noncontiguous data.

Derived data type

MPI derived types are constructed from primitive MPI data types (`mpi_integer, mpi_real, mpi_double_precision,` ...)

Definition: a derived data type is an opaque object that specifies a sequence of primitive types $t\_0, t\_1, ..., t\_n - 1$ and a sequence of displacements for each primitive type $d\_0, d\_1, ..., d\_n - 1$

- The *type map* of the derived data type is defined as follows.

$$\text{typemap} = \{(t\_0, d\_0), (t\_1, d\_1), ..., (t\_n - 1, d\_n - 1)\}$$

- The *type signature* of the derived data type is defined as follows.

$$\text{typesig} = \{t\_0, t\_1, ..., t\_n - 1\}$$

Observe that the type map together with the address of the message buffer specifies a communication buffer that consists of n entries where the i^{th} entry is at address $buff + d\_i$ and has type t_i.

Here are the type signatures of some common mpi types.

```
mpi_integer:           {integer, 0}
mpi_real:              {real, 0}
mpi_double_precision: {real8, 0}
```

Applications to scientific computing

Sending a row of a matrix

```
integer, parameters :: n=128
double precision    :: A(n,n)
```

We want to create a type map for a row in A;

$$\text{typemap} = \{(real8, 0), (real8, n), (real8, 2n), ..., (real, (n-1)n\}$$

Relevant routines

- `mpi_type_size(datatype, size, ierror)`; 'size' returns the size of the data type in bytes
- `mpi_type_extent(datatype, extent, ierror)`; deprecated feature
- `mpi_get_extent(datatype, lb, extent, ierror)`; returns the lower bound and extent of datatype; the upper bound = lb + extent

Constructors

There are 8 constructors of MPI data types. Each one becomes more and more general. Most general is mpi_type_create_struct(). You can acutally use this one to create all the others, but you should use the simplest available since the are typically optimized