

Message-Passing Programming

Memory allocation and ordering

Reusing this material



This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

http://creativecommons.org/licenses/by-nc-sa/4.0/deed.en_US

This means you are free to copy and redistribute the material and adapt and build on the material under the following terms: You must give appropriate credit, provide a link to the license and indicate if changes were made. If you adapt or build on the material you must distribute your work under the same license as the original.

Note that this presentation contains images owned by others. Please seek their permission before reusing these images.

Fortran array syntax

- MPI derived types enable strided data to be sent/received
 - no explicit copy in/out required
- For Fortran
 - why not use Fortran array syntax?
- Some subtleties for non-blocking operations
 - see notes on Learn

Array Layout in Memory

C: $\mathbf{x}[16]$

F: $\mathbf{x}(16)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----

C: $\mathbf{x}[4][4]$

F: $\mathbf{x}(4, 4)$

4	8	12	16
3	7	11	15
2	6	10	14
1	5	9	13

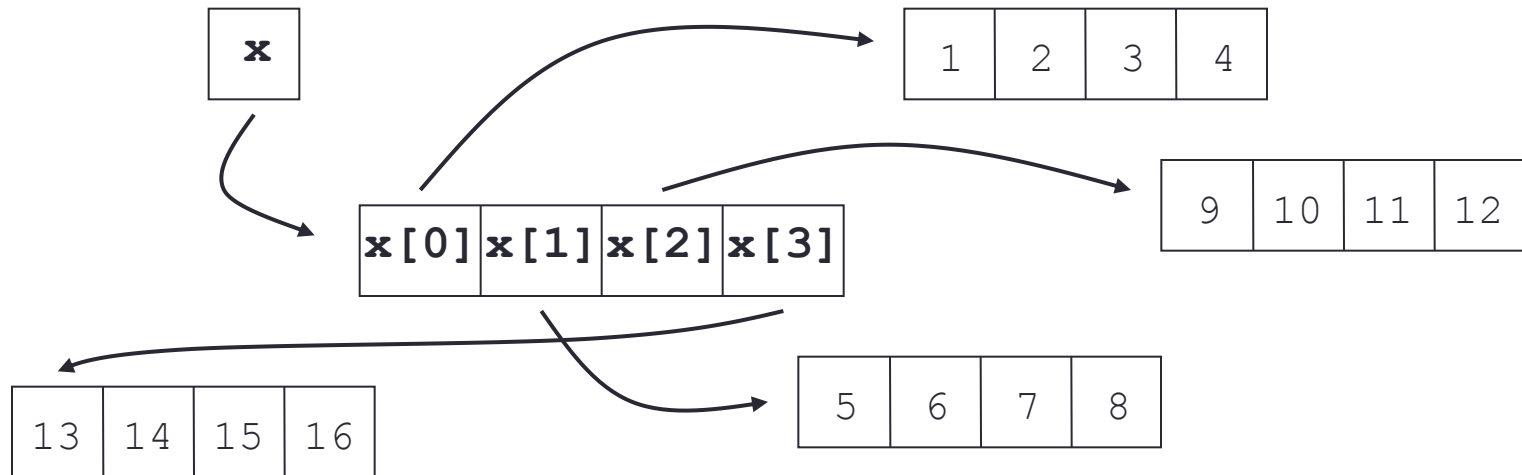


13	14	15	16
9	10	11	12
5	6	7	8
1	2	3	4

- Data is contiguous in memory
 - different conventions in C and Fortran
 - for statically allocated C arrays $\mathbf{x} == \&\mathbf{x}[0][0]$

Aside: Dynamic Arrays in C

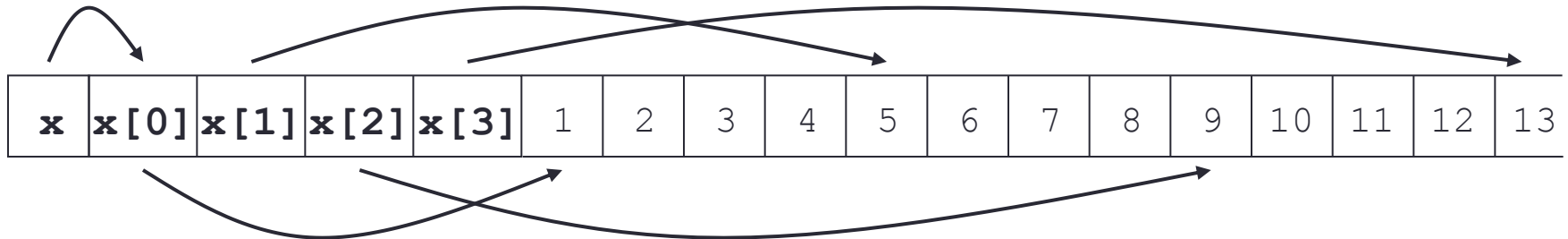
```
float **x = (float **) malloc(4, sizeof(float *));  
  
for (i=0; i < 4; i++)  
{  
    x[i] = (float *) malloc(4, sizeof(float));  
}
```



- Data non-contiguous, and `x != &x[0][0]`
 - cannot use regular templates such as vector datatypes
 - cannot pass `x` to any MPI routine

Arralloc

```
float **x = (float **) arralloc(sizeof(float), 2, 4, 4);  
/* do some work */  
free((void *) x);
```



- Data is now contiguous, but still **x** **!=** **&x[0][0]**
 - can now use regular template such as vector datatype
 - must pass **&x[0][0]** (start of contiguous data) to MPI routines
 - see **MPP-arralloc.tar** for example of use in practice
- Will illustrate all calls using **&x[i][j]** syntax
 - correct for both static and (contiguously allocated) dynamic arrays