

Exercise 2

Towards a solution

Paul Burton

Paul.Burton@ecmwf.int

General Guidance

- Break it into manageable pieces to deal with
 - Already nicely broken down into neat subroutines!
- Look at the data structures
 - How are you going to split between processors?
- Don't cheat
 - Try and work it out for yourself before you look through the rest of these slides!

Parallel Initialisation

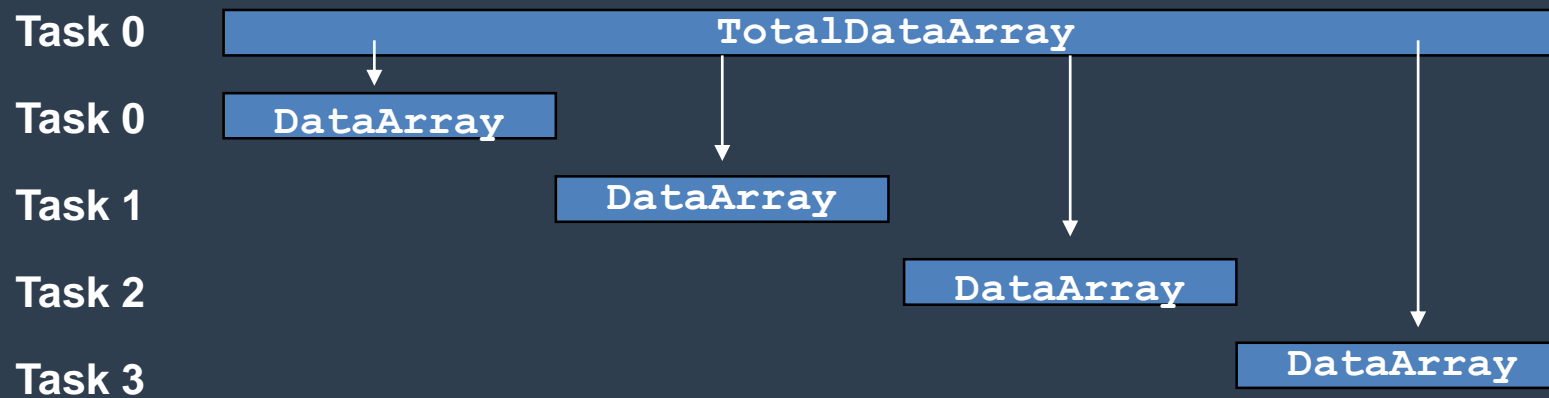
- Need to find out from MPI:
 - How many processors? (`NTasks`)
 - `CALL MPI_COMM_SIZE(MPI_COMM_WORLD,NTasks,ierror)`
 - What is my ID/Rank? (`MyTask`)
 - `CALL MPI_COMM_RANK(MPI_COMM_WORLD,MyTask,ierror)`
 - Who are my neighbours?
 - `MyNeighbourLeft=MyTask-1`
 - `MyNeighbourRight=MyTask+1`
 - Don't forget the wrap around, so it's a bit different for `MyTask=0` and `MyTask=NTasks-1`
 - Calculate `NPointsPerTask`

Model_Driver

- No longer with `npoints` (Total number of points)
 - Use `NPointsPerTask` (from `Parallel_Info_Mod`)

Read_Data

- Read all the data on Task 0
 - Need some logic to select the right task
 - We'll need a temporary array to hold the data on task 0
- Then scatter the data from Task 0 to all the tasks
 - Could use SEND/RECV
 - Easier to use `MPI_SCATTER`



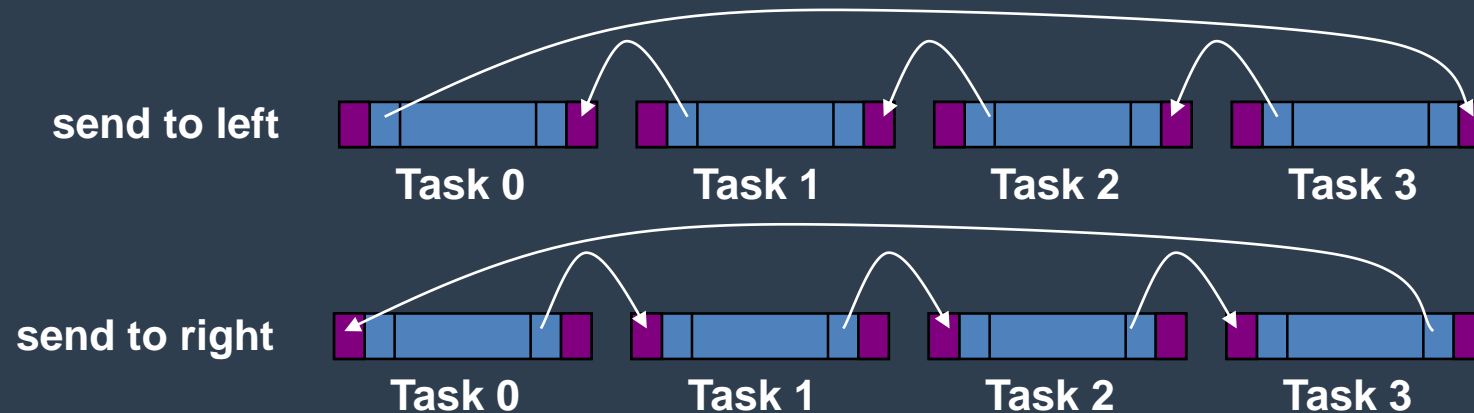
Sum_Data

- First calculate local sum
- Then add together all the local sums
 - Put the result on task 0
 - Could have all tasks sending local sum to task 0
 - Task 0 would then add these up
 - Better solution is to use `MPI_REDUCE`
 - Which does it all for you (efficiently hopefully!)



Finite_Difference

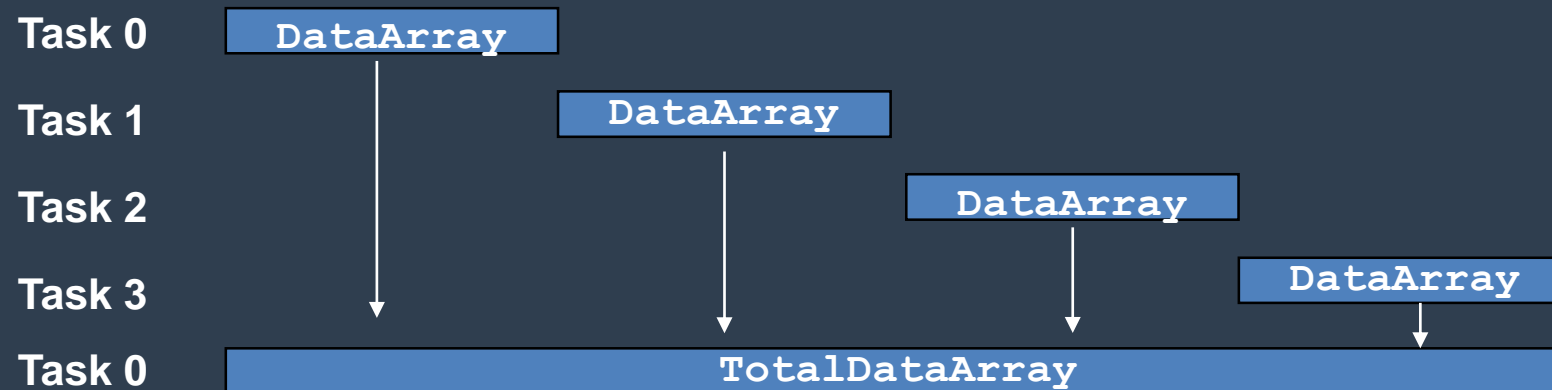
- Copy DataArray to OldData
 - But overdimension OldData (0:npoints+1)
 - We'll use the extra points at start and end as copies of points from the neighbouring tasks
- Communication
 - Could use SEND / RECV – but need to avoid blocking
 - Easier to use SENDRECV



- Do calculation from 1 to npoints

Write_Data

- Reverse of Read_Data
- Collect all the data onto Task 0
 - We'll need a temporary array to hold the data on task 0
- Gather the data from the tasks to Task 0
 - Could use `SEND/RECV`
 - Easier to use `MPI_Gather`



- And then write to disk on Task 0