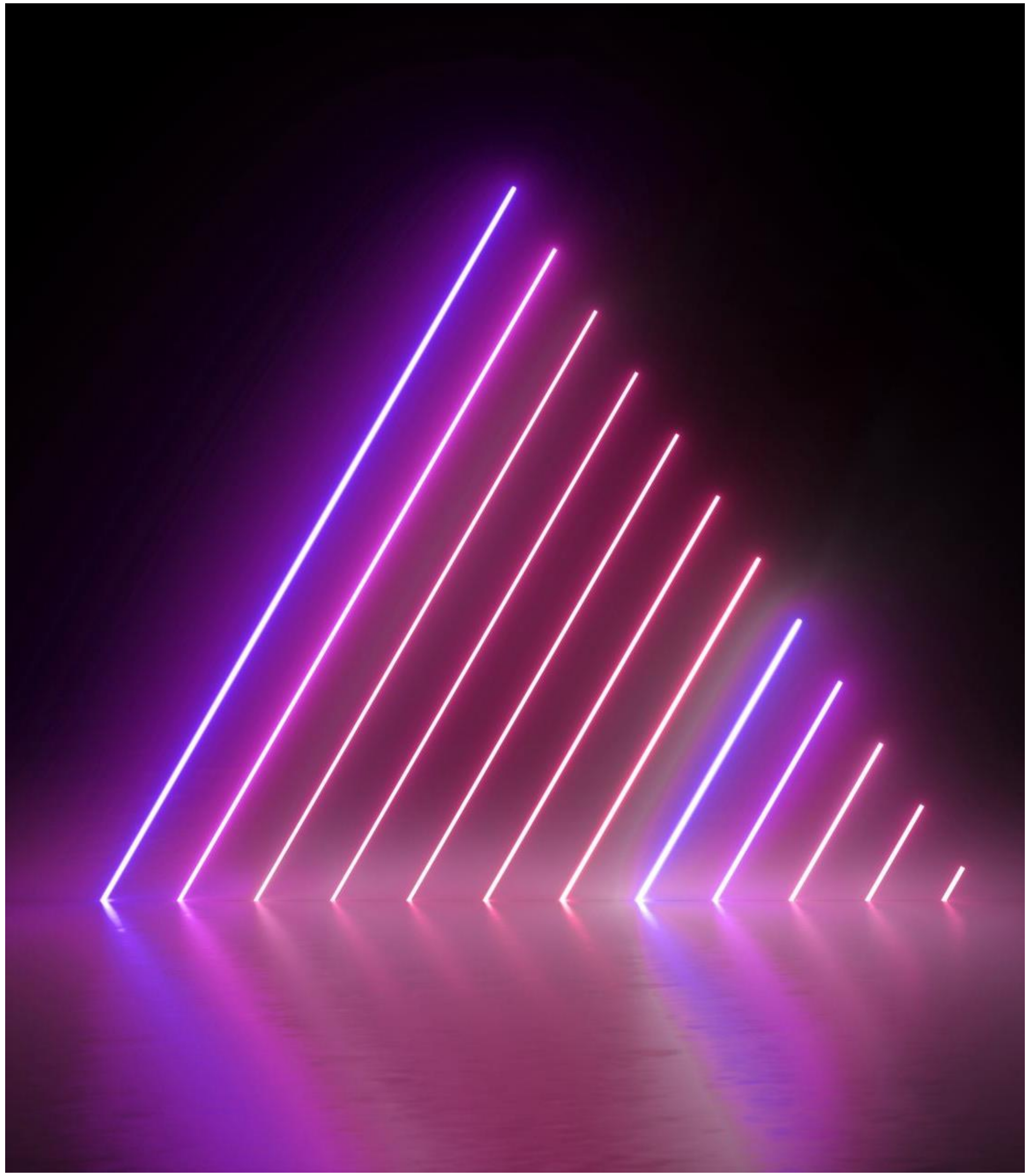
SOME ADVICE

ASSIGNMENT 1



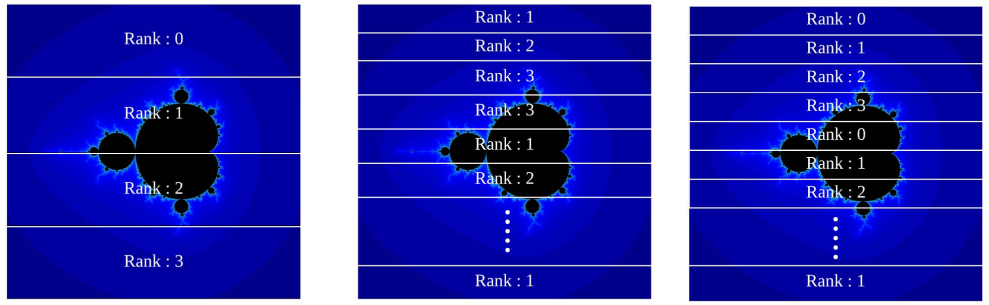
TA: Jo Inge Arnes

FROM THE ASSIGNMENT TEXT

Use some additional technique to solve the assignment other than just processing rows or columns by MPI workers. Some approaches that could be used are as follow: create a heat map of most demanding parts to balance the workload for each node, alternate row segmen- tation [4], apply First Come First Served approach [4] etc. If you are unsure about the additional technique please discuss it with TAs.

TWO TYPES OF PERFORMANCE ENHANCEMENTS

- Both affect performance
  1. Static vs dynamic taks scheduling
  2. How you partition the area
- You should ideally compare only one change at a time
- Two static versions and one dynamic will be enough for this
  - Based on partitioning in paper 4

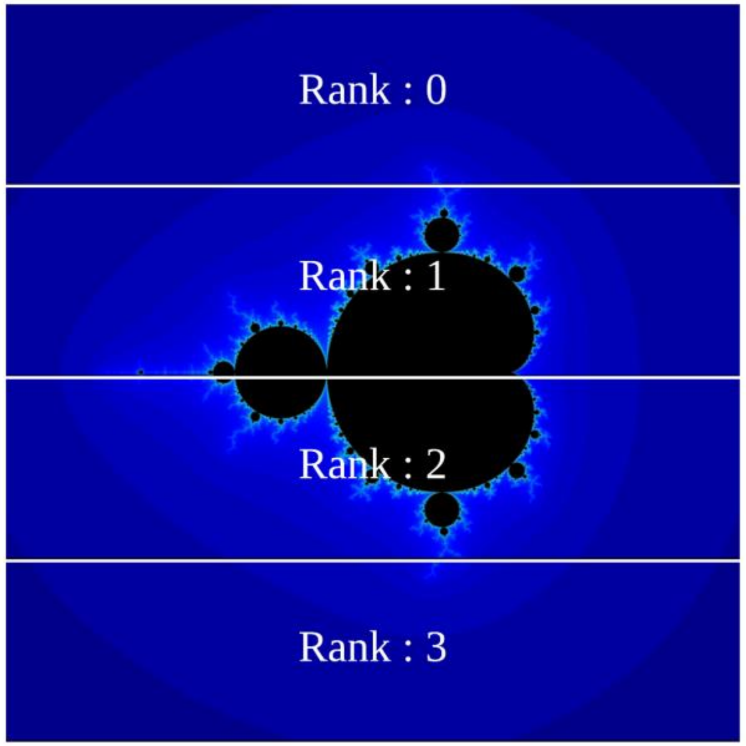


(a) Naive : Row Based Partition Scheme

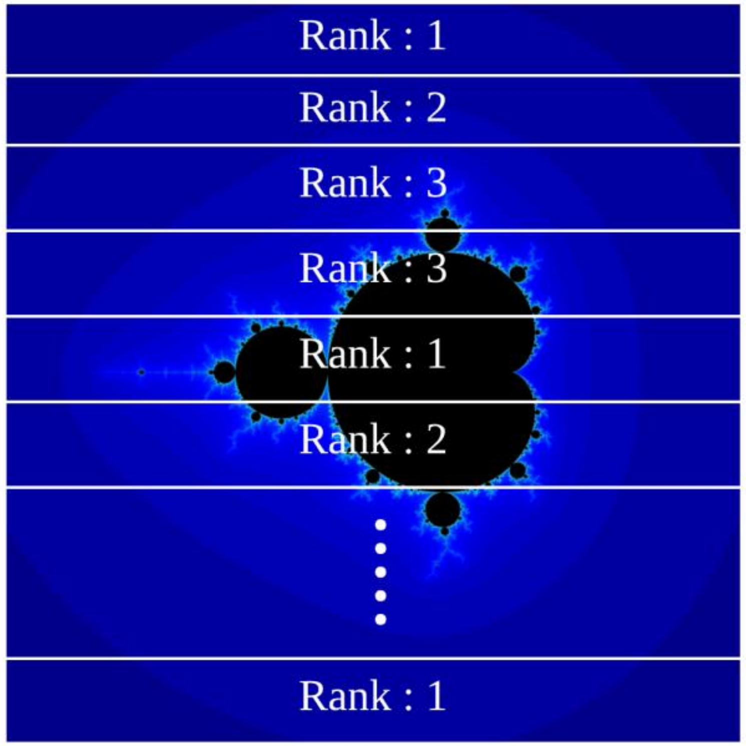
(b) First Come First Served : Row Based Partition Scheme

(c) Alternating : Row Based Partition Scheme

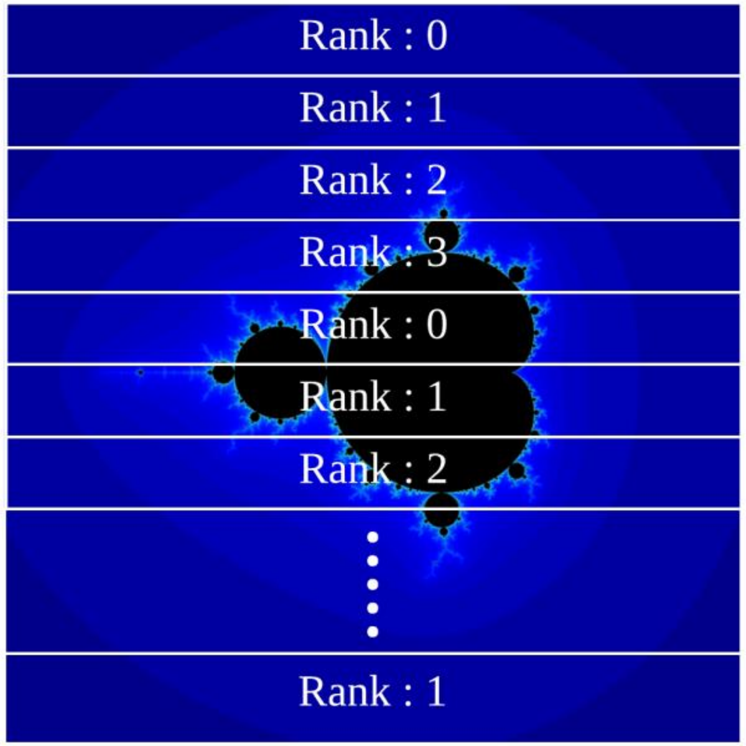
Fig. 2: Partition Schemes when $N = 4$



(a) Naive : Row Based Partition Scheme



(b) First Come First Served : Row Based Partition Scheme

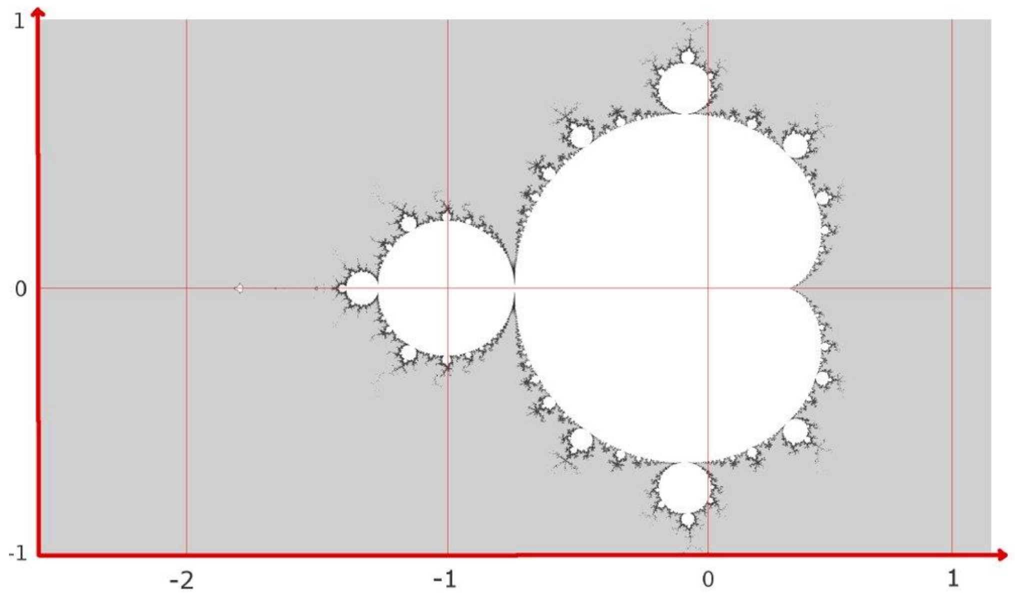


(c) Alternating : Row Based Partition Scheme

Fig. 2: Partition Schemes when $N = 4$

ESTIMATION OF COMPUTATIONAL DEMANDS OF DIFFERENT AREAS

- More advanced alternative
- Some areas require far fewer iterations compared to others.
- By partitioning and computing at a low resolution, the demand of different areas can be estimated
- The work can then be scheduled according to these estimates



FROM ASSIGNMENT TEXT

Remember to take into consideration the hardware you are using.

Hint: the MPI cluster is heterogeneous.

DIFFERENT HARDWARE ON NODES

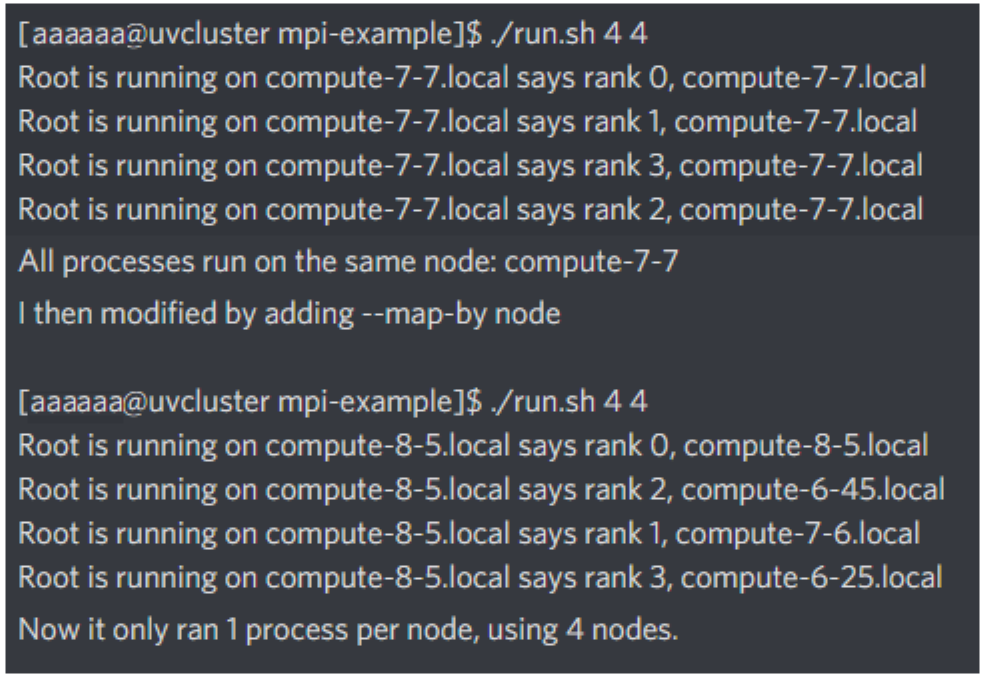
- To see more details for a compute node
  - From the front-end node ssh into a compute node
  - Use the command *lscpu*
- Or run the script for listing nodes on the front-end node. Can combine with grep. Example:

```
/share/apps/ifi/list-cluster-static.sh | grep compute-7-0
```

- Also, some nodes accept more than one process others only accept one

ONE PROCESS PER NODE

- The command *mpirun* (or *mpiexec*) has as parameter:
  `--map-by node`



```
[aaaaaa@uvcluster mpi-example]$ ./run.sh 4 4
Root is running on compute-7-7.local says rank 0, compute-7-7.local
Root is running on compute-7-7.local says rank 1, compute-7-7.local
Root is running on compute-7-7.local says rank 3, compute-7-7.local
Root is running on compute-7-7.local says rank 2, compute-7-7.local
```

All processes run on the same node: compute-7-7

I then modified by adding --map-by node

```
[aaaaaa@uvcluster mpi-example]$ ./run.sh 4 4
Root is running on compute-8-5.local says rank 0, compute-8-5.local
Root is running on compute-8-5.local says rank 2, compute-6-45.local
Root is running on compute-8-5.local says rank 1, compute-7-6.local
Root is running on compute-8-5.local says rank 3, compute-6-25.local
```

Now it only ran 1 process per node, using 4 nodes.