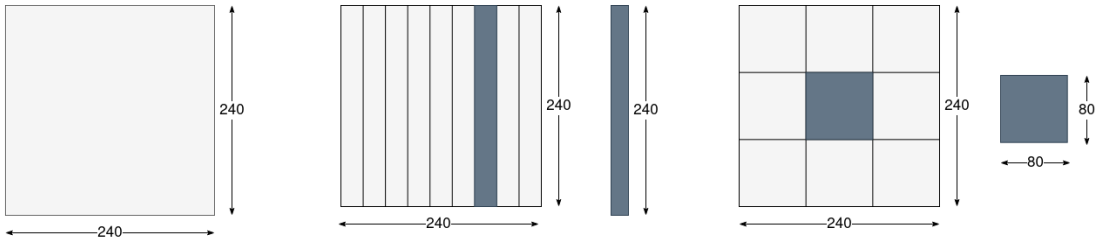
1 Task

1.1 Comparison of 1D and 2D decomposition

Describe the advantages/disadvantages of a two-dimensional decomposition (compared to a one-dimensional decomposition).

On the first look it seems that one advantage of the 1D decomposition has only to send to two neighbors, which seems less work than sending to all 4 neighbors as in the 2D case. But when looking at the actual data that has to be send and the time this takes, it is actally an advantage of the 2D decomposition with the scaling of the resolution. A good parameter to use is the speedup, is calculated as the runtime T_s divided by the parallel runtime T_p. $S = \frac{T_s}{T_p}$, this tells us how much the runtime changes with the inclusion of more processes. All in all in can be said that the scaling behavior between 1D decomposition 2D decomposition is significantly different, as the cutting surface of the 2D subdomains is smaller than in the 1D case. But this will not affect the speedup much for small numbers of CPU cores but for large numbers.

To make this clear we can look at an example, take a square with a resolution of 250 and calculate 100 timesteps. We deonte t_s as the time it takes for the calculation of one itteration for one node and t_p as the time it takes to communicate to other cells and lets assume they both take one time unit.



In the figure above a domain with resolution 240 is illustrated, the full domain can be seen on the left. If one would choose a 1D decomposition, the subdomains would look like the blue rectange in the center, withe the cuttung surface that needed to be communicated beeing 2x240 = 480. On the right the 2D decomposition can be seen. Here the edge of the subdomain is 4x80 = 320.

MIP processes	1D Speedup	2D Speedup
5	4.8	4.97
25	20.83	24.38
125	62.5	110.96
250	83.33	199.53

When looking at the tabel it can be seen, that for the 2D decomposition the speedup scales much better with resolution as the cutting surface of the domains in 1D stays the same for every resolution, but for 2D the cutting surface gets smaller.

Reference : https://alexander.vondrous.de/?p=7

1.2 Effect of decomposition on numercal solution

Discuss if the decomposition of the domain changes the order of computations performed during a single Jacobi iteration - i.e., if you expect a numerically identical result after each iteration, or not.

As the decomposition just seperates the domain in smaller subdomains but doesnt change the actual values and the subdomain boundaries get updated with their neigbouring values every itteration. We expect that the result after each itteration doesnt deviate numerically from the calculation without subdomains, up to machine precision. If one would update the ghost values less frequent we expect the error to get bigger.

1.3 Depth of ghost layer

A generalization of the ghost layer approach would be to set the width of the ghost layer that is exchanged as a parameter W of the decomposition. This allows to perform W independent iterations before a communication of the ghost layers has to happen. Comment in which situation (w.r.t the available bandwidth or latency between MPI-processes) multiple independent iterations are potentially advantageous.

With the approach of a ghost layer with depth one, one would have to update the subdomain boundaries at each step and therefor send many messages with a small size. This is a large communication overhead. If we generalization this approach as desribed in the question to a ghost layer with a width of W, this can be adressed, but the number of ghost layers also corresponds to the accuracy of the nummerical methods.

The idea behing this generalisation of the ghost layers, is the increase in message volume for each update. This means with a deeper ghost layers the domains have to be updated less often, for example if W = 6 the ghost layers only have to be updated once every five time steps, with six layers beeing exchanged at once. So with less frequent updates smaller messages are grouped into bigger messages, thus achieve higher communication bandwidth and lower communication latency.

This can be advantageous if the available bandwidth enables it. As we are working on the IUE-cluster a bandwidth of 10Gbit/s is provieded this could be utilied with this method. Also if latency reducion is the main objective changing the width of the ghost layers can be a good way to start. One disadvantag of a bigger Q is, that the method uses more memory and computation that the convential method, but in comparison to the total memory needed this is a small difference.

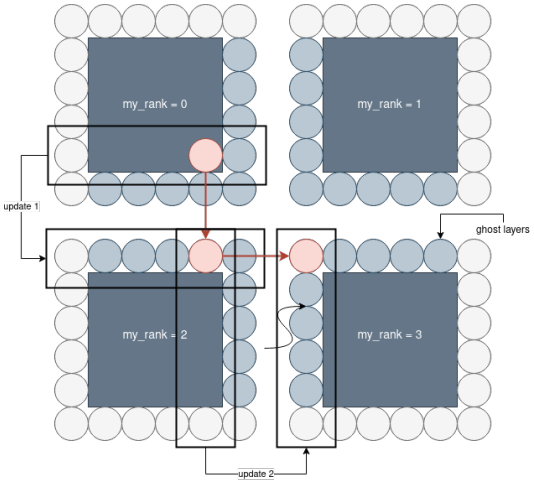
Reference: DOI: 10.1109/SC.2001.10042

1.4 Exchange with diagional neighbors

Assume a ghost layer with width W=1 (this is what you will later implement) and discuss if a data exchange between parts of the domain which are "diagonal neighbors" is required assuming a "5-point star-shaped stencil".

As we are working with a "5-point star-shaped stencil" the diagonal elements are never needet for an itteration of the boundary layers. Only the direct neighbors in North, South, West, East

direction are needed. Additionally if the communication to the ghost layers, as implied by the task description and the figure below, includes all elements in one row/collumn, it manages to exchange data with the "diagonal neighbors" aswell in two steps. This is illustrated in the figure below.



1.5 IUE-cluster

How big is the sum of all L2 caches for 2 nodes of the IUE-cluster

When looking at the stats of the IUE-cluster, two tyoes of computer node are available, 10x regular computer node and 2x fat computer nodes. Both types are as desibed on the officel website they are each equipped with 2x INTEL Xeon Gold 6248, 2.5GHz, 20C/40T. [1] Each processor has a L2 cache of 16 MB [2] so all together for 2 nodes and 2 INTEL each it sums up to 64 MB total.

Reference:

1 https://www.iue.tuwien.ac.at/research/computing-infrastructure/

2 https://www.cpubenchmark.net