Turbulent Flow Simulation on HPC Systems
Department of Mechanics and Informatics
Technical University of Munich

Worksheet 2

Haotian Ji, Siva Karthikeya Mandarapu, Hari Surya Charan Mudragada, and Matilde Tozzi

December, 2023

Abstract — This is a short abstract summarizing the main points of your article.

1 Parallelization (30%)

We looked at the `ParallelBoundaryIterator`, which operates on the border of a process domain and operates separately on the top, bottom, left and right faces (in 3D, on the front and back faces too). The stencil used by the iterator needs to have the functions `applyXWall`, where `X` is the various sides respectively. We decided to use this iterator and a similar structure for our stencils `PressureBufferFillStencil`, `PressureBufferReadStencil`, `VelocityBufferFillStencil` and `VelocityBufferReadStencil`.

We then integrated these operations in `PetscParallelManager`, that uses the methods `communicatePressure` and `communicateVelocity` to call `MPI_Sendrecv`. The calls to this class were added in the class `Simulation`.

2 Scaling and Efficiency (20%)

The setting for each test scenario can be found in the Appendix. As suggested, we also switched off the VTK output because it is not parallelized.

2.1 Cavity 2D

The execution time without `mpirun` is 16250192ns. This is the situation with different parallel domains:

Parallel Domain	Time
1x1x1	**14985779ns**
2x1x1	**14690334ns**
1x2x1	50765780ns
2x2x1	1019181815ns
4x2x1	2389953446ns
2x4x1	11321946305ns
4x4x1	739662694867ns

2.2 Cavity 3D

Parallel Domain	Time
Sequential	366442960ns
1x1x1	462171744ns
2x2x2	**134857738ns**
3x3x3	428625311ns

2.3 Channel 2D

Parallel Domain	Time
Sequential	35474301ns
1x1x1	43235603ns
2x1x1	50968602ns
1x2x1	74672623ns
2x2x1	71535312ns
4x2x1	72784482ns
2x4x1	86987863ns
4x4x1	2636085474ns

2.4 Channel 3D

Parallel Domain	Time
Sequential	1440966212ns
1x1x1	1762966207ns
2x2x2	**574001100ns**
3x3x3	5066212560ns

It looks like the time increases instead of decreasing in most cases: it may be that on a small scale the parallelization overhead is bigger than the possible improvement. It is also worth noting that for both 3D cases the 2x2x2 parallel domain obtains the best results by far, so probably also the divisibility of the simulation domain into the parallel domains plays a non-negligible role.

For this reason, we consider a wider domain, for example 50x60 in the 2D Channel scenario

Parallel Domain	Time
Sequential	4058563854ns
1x1x1	4852222103ns
2x1x1	**3428565005ns**
1x2x1	4996924073ns
2x2x1	**3112960876ns**
4x2x1	**2312658368ns**
2x4x1	**3182985333ns**
4x4x1	35063634635ns

It is quite clear here, especially considering the behaviour of the 4x4x1 domain, that there is a point after which the parallelization becomes way slower than the sequential version. If we take a much bigger domain (100x200) and a short time (0.01):

Parallel Domain	Time
Sequential	750595797ns
1x1x1	859073348ns
2x2x1	1955024379ns
3x3x1	1205773137ns
4x4x1	8021217360ns

If we consider the 3D Channel with a 100x50x100 spatial domain and 4x3x4 parallel domain we get 32397910983ns against 107381417991ns of sequential time, that is, 3.3 times faster. This is clearly still not a strong scaling, because we are using 48 processors.

3 Implementation Turbulence Modeling (30%)

4 Testing (5%)

5 Flow Physics (15%)

Appendix

Scaling test configurations

Cavity2D

```
  <?xml version="1.0" encoding="utf-8"?>
<configuration>
    <flow Re="500" />
    <simulation finalTime="0.5" >
        <type>dns</type>
        <scenario>cavity</scenario>
    </simulation>
    <timestep dt="1" tau="0.5" />
    <solver gamma="0.5" />
    <geometry dim="2"
      lengthX="1.0" lengthY="1.0" lengthZ="1.0"
      sizeX="20" sizeY="10" sizeZ="20"
    >
      <mesh>uniform</mesh>
    </geometry>
    <environment gx="0" gy="0" gz="0" />
    <walls>
        <left>
            <vector x="0" y="0" z="0" />
        </left>
        <right>
            <vector x="0" y="0" z="0" />
        </right>
        <top>
            <vector x="1" y="0" z="0" />
        </top>
        <bottom>
            <vector x="0" y="0" z="0" />
        </bottom>
        <front>
            <vector x="0" y="0" z="0" />
        </front>
        <back>
            <vector x="0" y="0" z="0" />
        </back>
    </walls>
    <vtk interval="0.01">Cavity2D</vtk>
    <stdOut interval="0.01" />
    <parallel numProcessorsX="1" numProcessorsY="1" numProcessorsZ="1" />
</configuration>
```

Cavity3D

```
  <?xml version="1.0" encoding="utf-8"?>
<configuration>
    <flow Re="500" />
    <simulation finalTime="0.5" >
```

```
        <type>dns</type>
        <scenario>cavity</scenario>
    </simulation>
    <timestep dt="1" tau="0.5" />
    <solver gamma="0.5" />
    <geometry dim="3"
      lengthX="1.0" lengthY="1.0" lengthZ="1.0"
      sizeX="20" sizeY="10" sizeZ="20"
    >
      <mesh>uniform</mesh>
    </geometry>
    <environment gx="0" gy="0" gz="0" />
    <walls>
        <left>
            <vector x="0" y="0" z="0" />
        </left>
        <right>
            <vector x="0" y="0" z="0" />
        </right>
        <top>
            <vector x="1" y="0" z="0" />
        </top>
        <bottom>
            <vector x="0" y="0" z="0" />
        </bottom>
        <front>
            <vector x="0" y="0" z="0" />
        </front>
        <back>
            <vector x="0" y="0" z="0" />
        </back>
    </walls>
    <vtk interval="0.01">Cavity3D</vtk>
    <stdOut interval="0.01" />
    <parallel numProcessorsX="1" numProcessorsY="1" numProcessorsZ="1" />
</configuration>
```