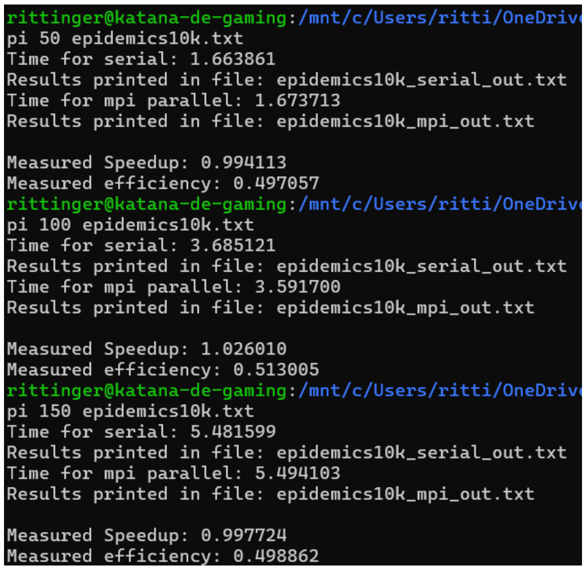
Results for epidemic simulation program using MPI

Rittinger Stefan

1) 10K people: 2 processes; 50, 100, 150, 200, 500 simulation time
   Dimension of grid: 110 x 110

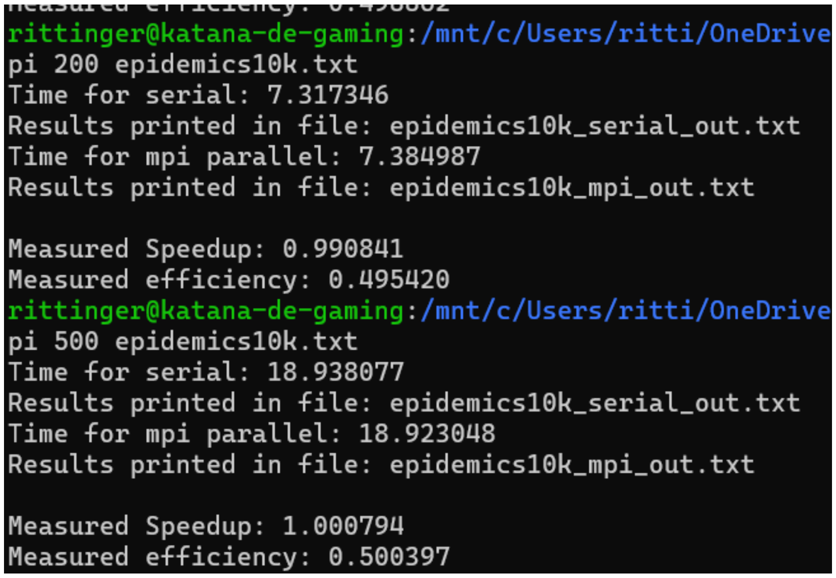


```
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDriv
pi 50 epidemics10k.txt
Time for serial: 1.663861
Results printed in file: epidemics10k_serial_out.txt
Time for mpi parallel: 1.673713
Results printed in file: epidemics10k_mpi_out.txt

Measured Speedup: 0.994113
Measured efficiency: 0.497057
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDriv
pi 100 epidemics10k.txt
Time for serial: 3.685121
Results printed in file: epidemics10k_serial_out.txt
Time for mpi parallel: 3.591700
Results printed in file: epidemics10k_mpi_out.txt

Measured Speedup: 1.026010
Measured efficiency: 0.513005
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDriv
pi 150 epidemics10k.txt
Time for serial: 5.481599
Results printed in file: epidemics10k_serial_out.txt
Time for mpi parallel: 5.494103
Results printed in file: epidemics10k_mpi_out.txt

Measured Speedup: 0.997724
Measured efficiency: 0.498862
```

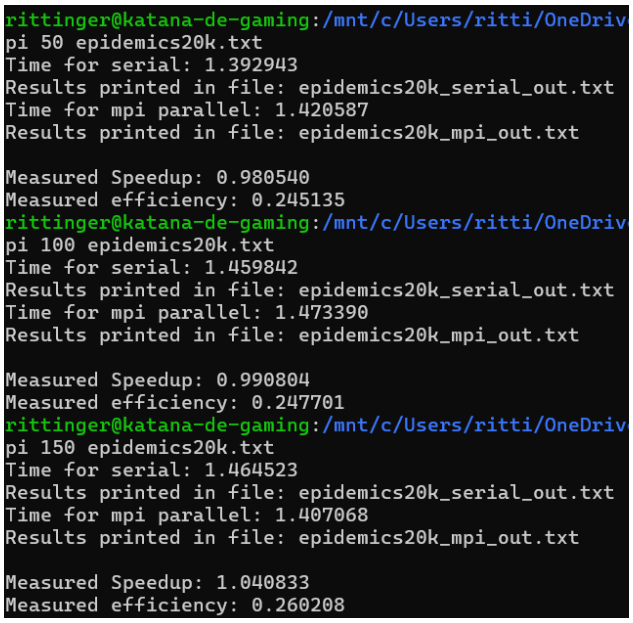


```
Measured efficiency: 0.498862
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDrive
pi 200 epidemics10k.txt
Time for serial: 7.317346
Results printed in file: epidemics10k_serial_out.txt
Time for mpi parallel: 7.384987
Results printed in file: epidemics10k_mpi_out.txt

Measured Speedup: 0.990841
Measured efficiency: 0.495420
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDrive
pi 500 epidemics10k.txt
Time for serial: 18.938077
Results printed in file: epidemics10k_serial_out.txt
Time for mpi parallel: 18.923048
Results printed in file: epidemics10k_mpi_out.txt

Measured Speedup: 1.000794
Measured efficiency: 0.500397
```

2) 20K: 4 processes; 50, 100, 150, 200, 500 simulation time
   Dimensions of grid: 30x30

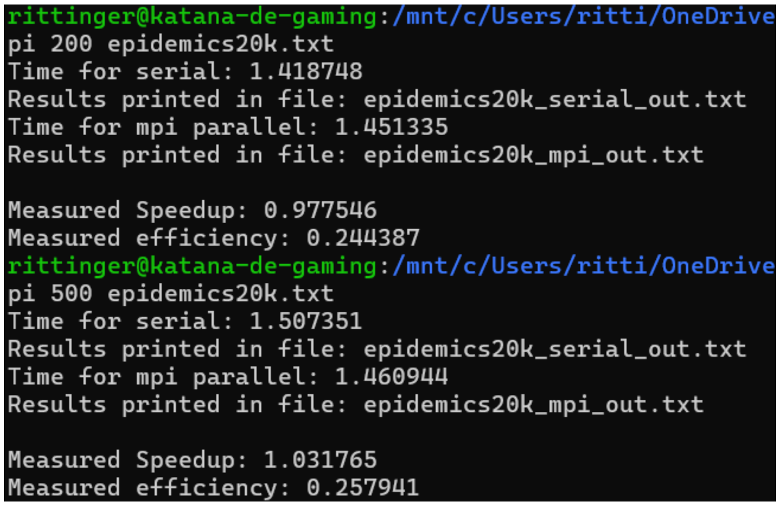


```
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDriv
pi 50 epidemics20k.txt
Time for serial: 1.392943
Results printed in file: epidemics20k_serial_out.txt
Time for mpi parallel: 1.420587
Results printed in file: epidemics20k_mpi_out.txt

Measured Speedup: 0.980540
Measured efficiency: 0.245135
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDriv
pi 100 epidemics20k.txt
Time for serial: 1.459842
Results printed in file: epidemics20k_serial_out.txt
Time for mpi parallel: 1.473390
Results printed in file: epidemics20k_mpi_out.txt

Measured Speedup: 0.990804
Measured efficiency: 0.247701
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDriv
pi 150 epidemics20k.txt
Time for serial: 1.464523
Results printed in file: epidemics20k_serial_out.txt
Time for mpi parallel: 1.407068
Results printed in file: epidemics20k_mpi_out.txt

Measured Speedup: 1.040833
Measured efficiency: 0.260208
```

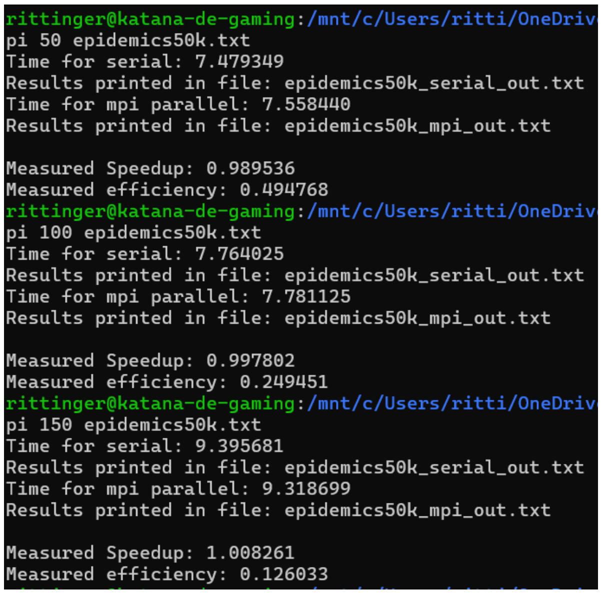


```
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDrive
pi 200 epidemics20k.txt
Time for serial: 1.418748
Results printed in file: epidemics20k_serial_out.txt
Time for mpi parallel: 1.451335
Results printed in file: epidemics20k_mpi_out.txt

Measured Speedup: 0.977546
Measured efficiency: 0.244387
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDrive
pi 500 epidemics20k.txt
Time for serial: 1.507351
Results printed in file: epidemics20k_serial_out.txt
Time for mpi parallel: 1.460944
Results printed in file: epidemics20k_mpi_out.txt

Measured Speedup: 1.031765
Measured efficiency: 0.257941
```

3) 50K people: varied number of processes for the sake of experimenting 50, 100, 150, 200, 500 simulation time
Dimension of grid: 75x75

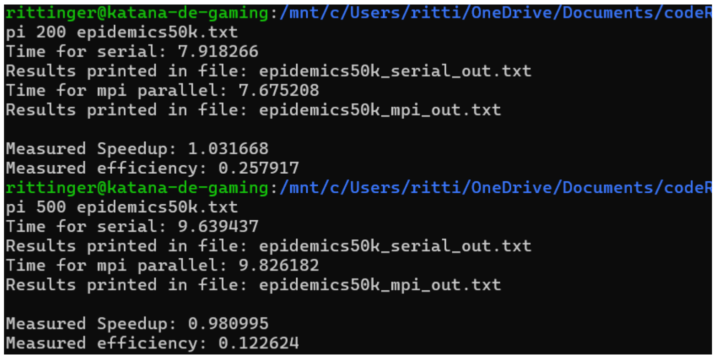


```
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDriv
pi 50 epidemics50k.txt
Time for serial: 7.479349
Results printed in file: epidemics50k_serial_out.txt
Time for mpi parallel: 7.558440
Results printed in file: epidemics50k_mpi_out.txt

Measured Speedup: 0.989536
Measured efficiency: 0.494768
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDriv
pi 100 epidemics50k.txt
Time for serial: 7.764025
Results printed in file: epidemics50k_serial_out.txt
Time for mpi parallel: 7.781125
Results printed in file: epidemics50k_mpi_out.txt

Measured Speedup: 0.997802
Measured efficiency: 0.249451
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDriv
pi 150 epidemics50k.txt
Time for serial: 9.395681
Results printed in file: epidemics50k_serial_out.txt
Time for mpi parallel: 9.318699
Results printed in file: epidemics50k_mpi_out.txt

Measured Speedup: 1.008261
Measured efficiency: 0.126033
```

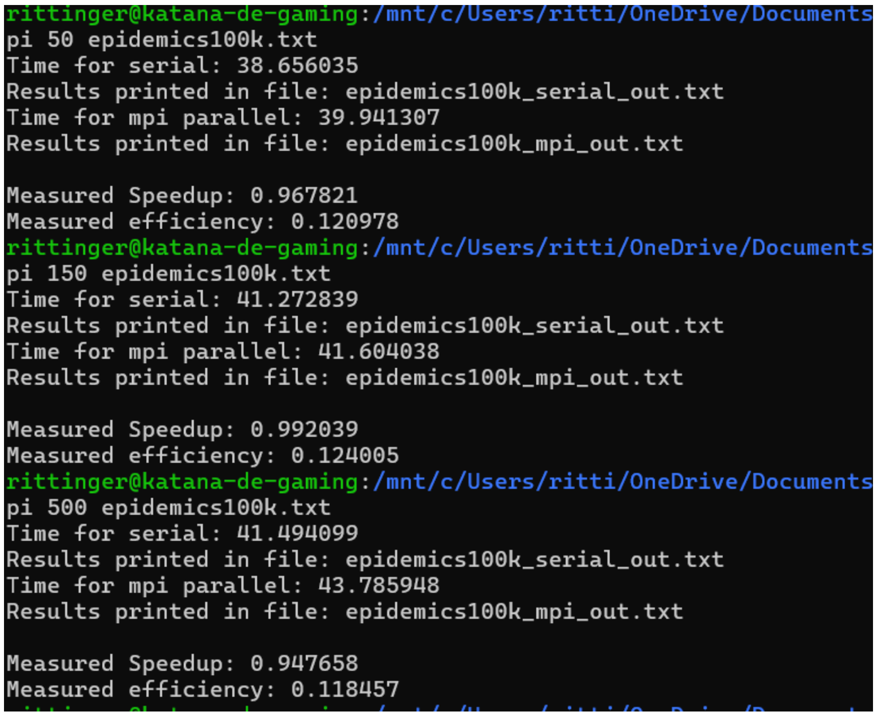


```
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDrive/Documents/codeR
pi 200 epidemics50k.txt
Time for serial: 7.918266
Results printed in file: epidemics50k_serial_out.txt
Time for mpi parallel: 7.675208
Results printed in file: epidemics50k_mpi_out.txt

Measured Speedup: 1.031668
Measured efficiency: 0.257917
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDrive/Documents/codeR
pi 500 epidemics50k.txt
Time for serial: 9.639437
Results printed in file: epidemics50k_serial_out.txt
Time for mpi parallel: 9.826182
Results printed in file: epidemics50k_mpi_out.txt

Measured Speedup: 0.980995
Measured efficiency: 0.122624
```

4) 100K people: 8 processes; 50, 150, 500 simulation time
Dimension of grid: 60x60



```
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDrive/Documents
pi 50 epidemics100k.txt
Time for serial: 38.656035
Results printed in file: epidemics100k_serial_out.txt
Time for mpi parallel: 39.941307
Results printed in file: epidemics100k_mpi_out.txt

Measured Speedup: 0.967821
Measured efficiency: 0.120978
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDrive/Documents
pi 150 epidemics100k.txt
Time for serial: 41.272839
Results printed in file: epidemics100k_serial_out.txt
Time for mpi parallel: 41.604038
Results printed in file: epidemics100k_mpi_out.txt

Measured Speedup: 0.992039
Measured efficiency: 0.124005
rittinger@katana-de-gaming:/mnt/c/Users/ritti/OneDrive/Documents
pi 500 epidemics100k.txt
Time for serial: 41.494099
Results printed in file: epidemics100k_serial_out.txt
Time for mpi parallel: 43.785948
Results printed in file: epidemics100k_mpi_out.txt

Measured Speedup: 0.947658
Measured efficiency: 0.118457
```

Discussion:

1) The MPI parallelization using the serial version of this program is difficult, hence the results being quite disastrous as far as performance goes. This is due to the fact that the step of infecting people, as it is implemented in my program, requires the whole array to be checked for each person. This is of course where all the time complexity stems from. A way to parallelize this step would require partitioning the array of people, while also sending the complete array of people to each process, each time step of the simulation. This of course would bring as much overhead as the current method, especially as the size of the person array increases. Additionally, using some redundant data structure to simplify the operation would also be difficult, as this would also have to be sent to all processes at each time step. In conclusion, parallelizing this problem works better with a shared memory system, where the array can be updated easier by simply updating the one in the shared memory.

2) There is additional overhead coming from the constant scatter and gather operations being performed at each time step. This could be eliminated by finding a method to share the initial array once and only gather at the end of the simulation, which is difficult due to the fact that iterations depend on previous results.