**CSCE659 HW2 Jacobi Iterative Solver for 2D Laplace Equation**

**Haiping Xue**

**UIN: 122006286**

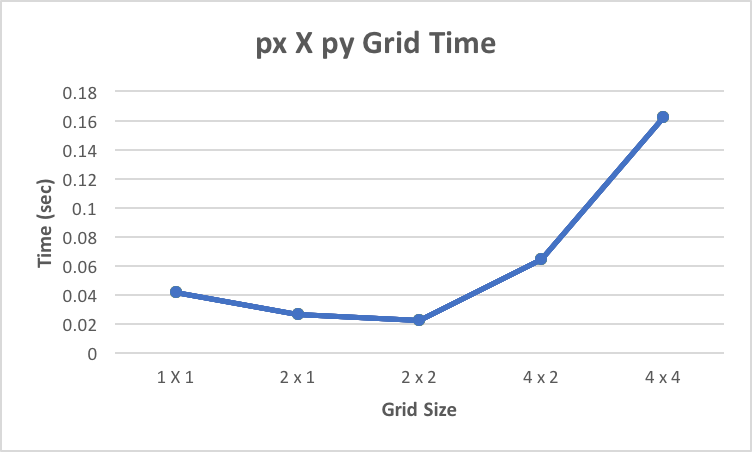
**Description of Changes in the code**

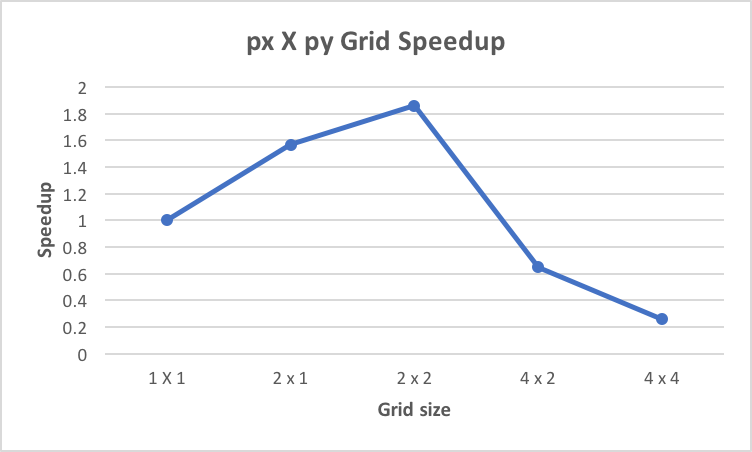
I added MPI send and receive functions for communication with top, below, left and right neighbor to places has insert code comment. I also added a MPI reduceAll function for normalization. The compilation follows the original code’s method.

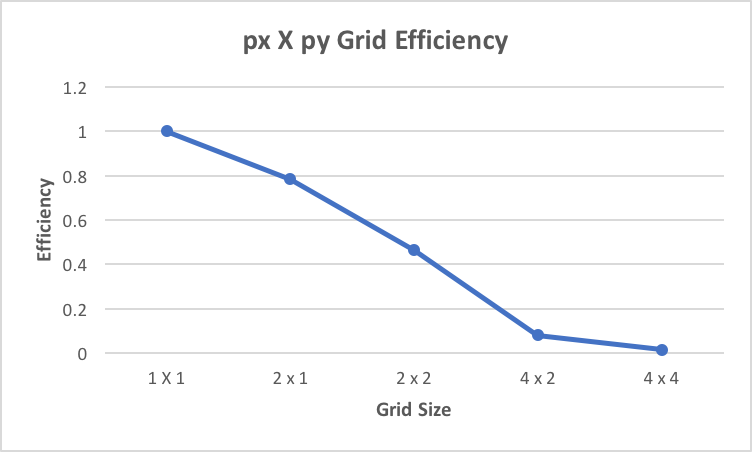
1. **Question 2**

**Experiment Set 1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Grid | Number of Processors | Times | Speedup | Efficiency |
| 1 X 1 | 1 | 0.041992 | 1 | 1 |
| 2 x 1 | 2 | 0.026815 | 1.565989185 | 0.782994593 |
| 2 x 2 | 4 | 0.022585 | 1.859287137 | 0.464821784 |
| 4 x 2 | 8 | 0.064688 | 0.649146673 | 0.081143334 |
| 4 x 4 | 16 | 0.162305 | 0.258722775 | 0.016170173 |

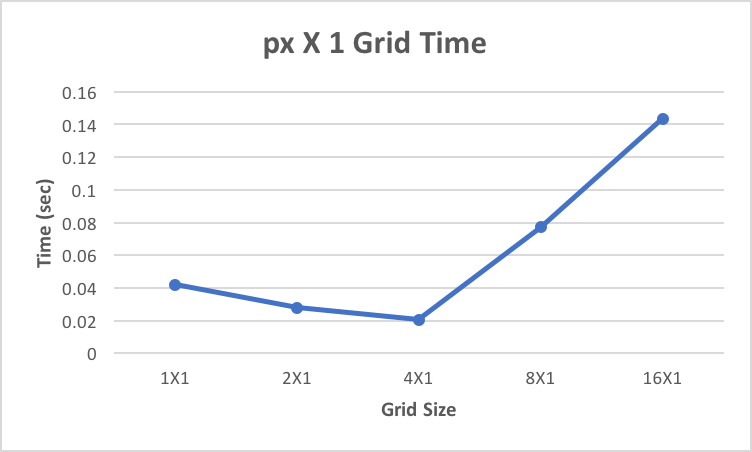


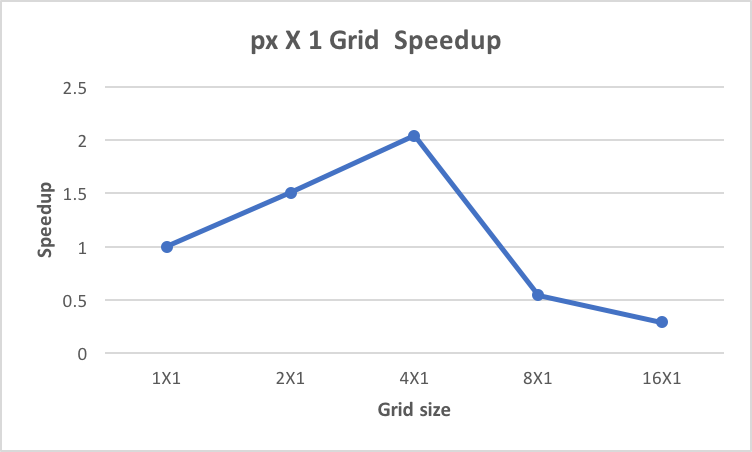
****

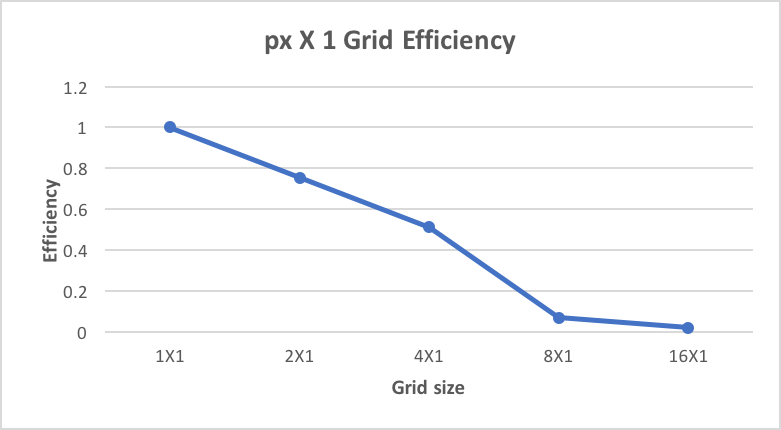
****

**Experiment Set 2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Grid | Number of Processors | Times | Speedup | Efficiency |
| 1X1 | 1 | 0.041884 | 1 | 1 |
| 2X1 | 2 | 0.027826 | 1.505210954 | 0.752605477 |
| 4X1 | 4 | 0.020522 | 2.040931683 | 0.510232921 |
| 8X1 | 8 | 0.07704 | 0.543665628 | 0.067958204 |
| 16X1 | 16 | 0.143445 | 0.291986476 | 0.018249155 |







**Conclusion**

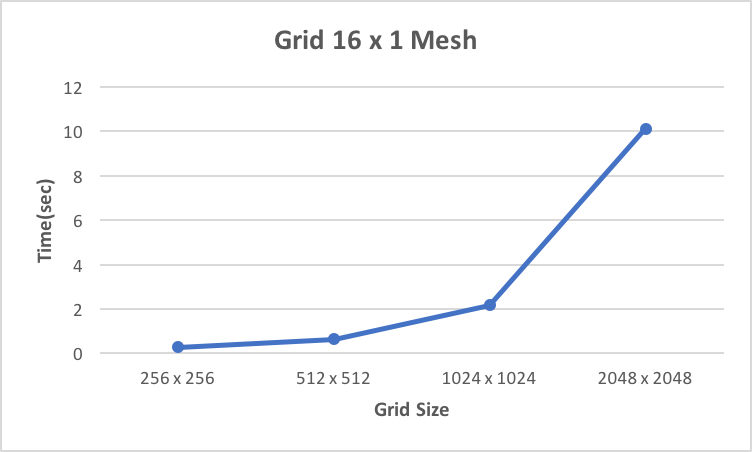
For this set of experiments, the py = 1 set’s performance is a little bit better than the (px , py) set. We can see that the communication cost for the program is growing as the number of processors is growing. After 4 processors for both 2X2 and 4X1 mesh, the speedup and efficiency of this program is dropping while the execution time is increasing rapidly due to the heavy communication among processes. Comparing pxX1 with pxXpy, the pxX1 is a little bit faster than pxXpy. Therefore, have 4 processes with 4X1 mesh will give my program the best performance. In addition to this, we can conclude that this parallel program does not work well with small-size meshes. Single process program is better for small-size mesh.

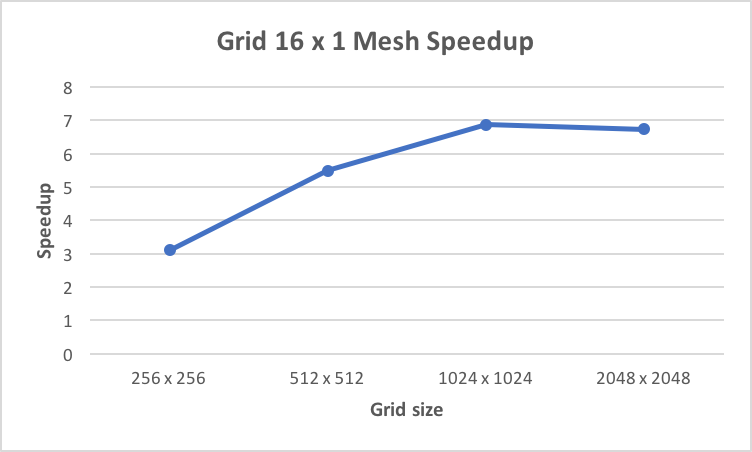
1. **Question 3**

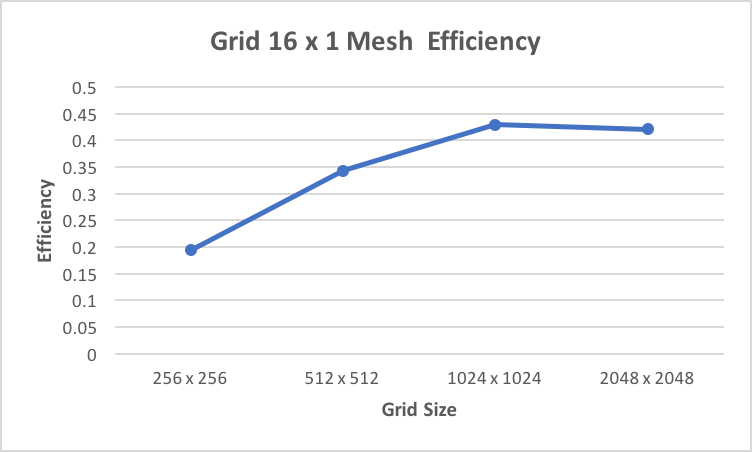
|  |  |  |
| --- | --- | --- |
| Baseline | Number of Processors | Times |
| 256 x 256 | 1 | 0.875919 |
| 512 x 512 | 1 | 3.481853 |
| 1024 x 1024 | 1 | 14.945813 |
| 2048 x 2048 | 1 | 68.063227 |

**Experiment Set 1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Grid 16 x 1 Mesh | Number of Processors | Times | Speedup | Efficiency |
| 256 x 256 | 16 | 0.282765 | 3.09769243 | 0.193605777 |
| 512 x 512 | 16 | 0.633846 | 5.493216018 | 0.343326001 |
| 1024 x 1024 | 16 | 2.17509 | 6.871353829 | 0.429459614 |
| 2048 x 2048 | 16 | 10.106299 | 6.73473316 | 0.420920822 |

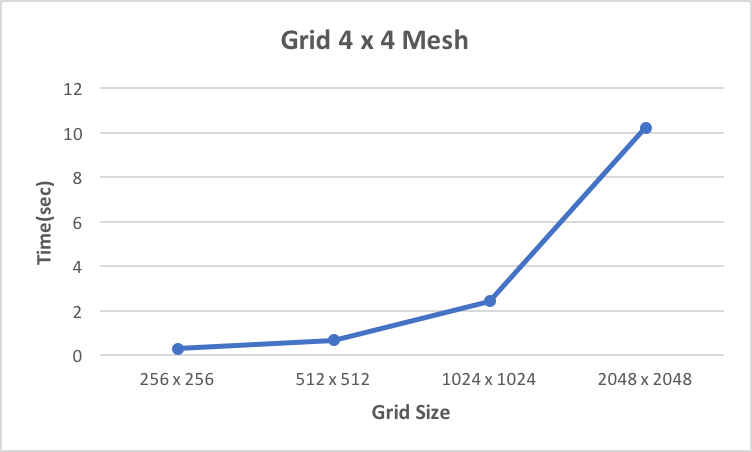


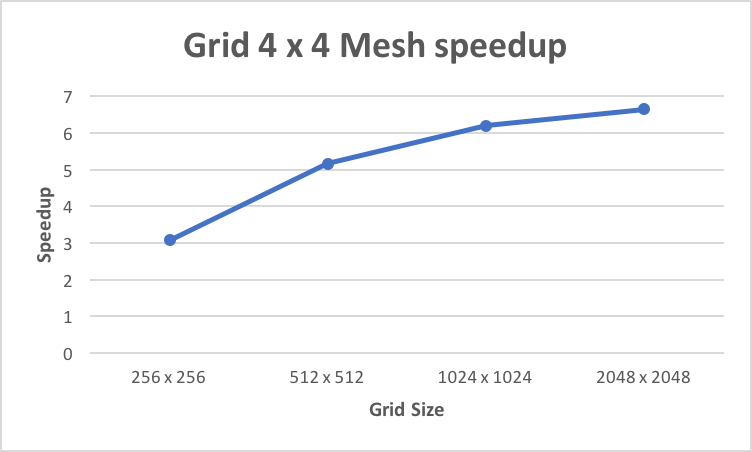
****

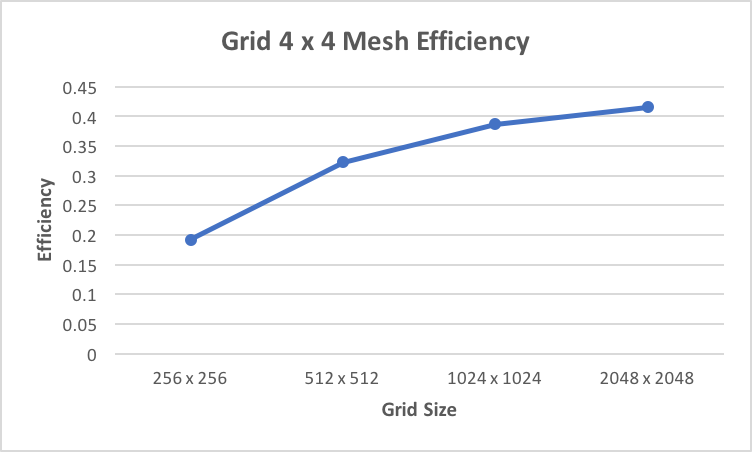
****

**Experiment Set 2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Grid 4 x 4 Mesh | Number of Processors | Times | Speedup | Efficiency |
| 256 x 256 | 16 | 0.285231 | 3.070910946 | 0.191931934 |
| 512 x 512 | 16 | 0.674523 | 5.161948518 | 0.322621782 |
| 1024 x 1024 | 16 | 2.41308 | 6.1936666 | 0.387104163 |
| 2048 x 2048 | 16 | 10.231867 | 6.652082851 | 0.415755178 |



****

****

**Conclusion**

For this set of experiments, I used one processor as the baseline. The execution time is increasing while we are using larger meshes. This makes sense since the computation size is increasing exponentially. As we can see that the speedup and efficiency of this program is increasing while we are making large meshes. This shows that the parallel program works well with large-size meshes and it can scale well. The speedup and efficiency of the program is increasing while the mesh size is growing. Comparing the pxXpy and pxX1, the execution time and performance is quite similar.