**Strong Scaling:**

From the graphs above, it can be seen that pure MPI with both synchronous and asynchronous communication scale strongly quite well. For each increase in processes, the speedup almost doubles. This trend occurs until 32 processes for both types of communication. With synchronous communication, speedup drops at the 960x960 and 1920x1920 matrix sizes but keeps increasing for a matrix size of 2880x2880. This likely stems from the fact that the communication overhead for the smaller matrix sizes begins to take more effect compared to the larger matrices. In asynchronous communication the large (2880x2880) matrix drops in speedup compared to the smaller ones. Both synchronous and asynchronous had very similar runtimes and speedups. See full results in Appendix A.

When threading inside the nodes was introduced with pthreads and OpenMPI the speedup was not remotely as good as pure MPI. This was because of our methodology in threading inside the nodes. Each time through the loop that iterated through the number of nodes the desired number of pthreads or OpenMP threads was created. For example, if there were four MPI nodes and eight threads, then eight threads were being created three times for a total of 24 threads being created. This caused a large amount of overhead that quickly overwhelmed any advantage of multithreading. However, when the number of total threads began to increase past 8 (1 MPI node with 8 threads) the speedup began to increase to reasonable levels. Unfortunately, pure runtime was much slower than just MPI. Theoretically, with a better implementation, the hybrid cases should match, if not better, the speedup of pure MPI.

Overall, the pure MPI versions had much better strong scaling, with synchronous communication MPI being the best, than the hybrid implementations.

In order to test strong scaling with pure MPI we ran the following commands:

mpirun -np 1 -N 1 -hostfile nodes (1 node with 1 process: 1 total)

mpirun -np 2 -N 2 -hostfile nodes (1 node with 2 processes 2 total)

mpirun -np 4 -N 4 -hostfile nodes (1 node with 4 processes: 4 total)

mpirun -np 8 -N 8 -hostfile nodes (1 node with 8 processes: 8 total)

mpirun -np 16 -N 8 -hostfile nodes (2 nodes with 8 processes 16 total)

mpirun -np 24 -N 8 -hostfile nodes (3 nodes with 8 processes 24 total)

mpirun -np 32 -N 8 -hostfile nodes (4 nodes with 8 processes: 32 total)

This ensured that each node was fully utilized and filled up before launching a new node.

When testing strong scaling with pthreads and OpenMPI, a similar pattern was followed:

mpirun -np 1 -hostfile nodes a.out 1 (1 node with 1 threads: 1 total)

mpirun -np 1 -hostfile nodes a.out 2 (1 node with 2 threads 2 total)

mpirun -np 1 -hostfile nodes a.out 4 (1 node with 4 threads: 4 total)

mpirun -np 1 -hostfile nodes a.out 8 (1 node with 8 threads: 8 total)

mpirun -np 2 -hostfile nodes a.out 8 (2 nodes with 8 threads: 16 total)

mpirun -np 3 -hostfile nodes a.out 8 (3 nodes with 8 threads: 24 total)

mpirun -np 4 -hostfile nodes a.out 8 (4 nodes with 8 threads: 32 total)

**Weak Scaling:**

From the graphs above, pure MPI has decent weak scaling properties. Gflops/thread (gigaflops/thread) remain relatively constant at about 100 until 16 processes. This occurs because after 8 processes (1 node with 8 processes) a new node is introduced (2 nodes with 8 processes each) and internode communication overhead starts to add up. After 16 processes Gflops/thread remains pretty constant despite adding more nodes. However, this obvious drop in Gflops/thread only occurs with synchronous communication. With asynchronous communication, Gflops/thread slowly decreases as more and more processes are introduced. This probably stems from the fact that each process must wait on the rest to ensure that the correct data has been received before continuing on. From these observations, it is reasonable to conclude that synchronous communication has better weak scaling properties, especially since synchronous communication levels out at higher Gflops/thread.

Weak scaling in the hybrid cases with MPI/pthreads and MPI/OpenMP is non-existent. Gflops/thread drop drastically in both cases as the number of processes is increased and level out at about 5 Gflops/thread at 8 threads. This is due to the same issue discussed in the strong scaling section where the thread creation overhead simply overwhelms any benefit in multithreading. Weak scaling in the hybrid cases is much worse than in pure MPI.

When testing weak scaling we used similar mpirun commands as we did in strong scaling to fill up a node with processes before adding another node, except we increased the dimension of the matrix proportionally to ensure that each node always had the same number of columns. For example, we wanted 480 columns per node, so for 1, 2, 4, and 8 processes we kept the matrix size at 480x480 since there was only one node. Then when we went to 16, 24, and 32 processes (2, 3, and 4 nodes), we increased the dimension to 960x960, 1920x1920, and 2880x2880 respectively. This made sure that each node always had 480 columns to work on.

Overall, the pure MPI implementations, with synchronous communication being the best, had better weak scaling properties than did the hybrid cases, but with a better implementation the hybrid cases should have marginally better weak scaling since internode communication overhead would be minimized.

For full data and results see Appendix B.

**Appendix A:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1a** | Runtime in seconds |  |  |  | Speedup |  |  |  |
| Threads | 960x960 | 1920x1920 | 2880x2880 |  | Threads | 960x960 | 1920x1920 | 2880x2880 |
| 1 | 8.37 | 67.8 | 229.91 |  | 1 | 1 | 1 | 1 |
| 2 | 4.6 | 36.15 | 123.27 |  | 2 | 1.819565217 | 1.875518672 | 1.865092886 |
| 4 | 2.32 | 18.57 | 63.64 |  | 4 | 3.607758621 | 3.651050081 | 3.612664991 |
| 8 | 1.16 | 9.31 | 31.47 |  | 8 | 7.215517241 | 7.282491944 | 7.305687957 |
| 16 | 1.06 | 8.11 | 26.45 |  | 16 | 7.896226415 | 8.360049322 | 8.692249527 |
| 24 | 0.77 | 5.83 | 18.41 |  | 24 | 10.87012987 | 11.62950257 | 12.48832156 |
| 32 | 1.61 | 4.33 | 26.45 |  | 32 | 5.198757764 | 15.65819861 | 8.692249527 |
| **1b** | Runtime in seconds |  |  |  | Speedup |  |  |  |
| Threads | 960x960 | 1920x1920 | 2880x2880 |  | Threads | 960x960 | 1920x1920 | 2880x2880 |
| 1 | 8.4 | 68.1 | 230.94 |  | 1 | 1 | 1 | 1 |
| 2 | 4.43 | 36.48 | 120.24 |  | 2 | 1.896162528 | 1.866776316 | 1.920658683 |
| 4 | 3.45 | 23.18 | 94.16 |  | 4 | 2.434782609 | 2.937877481 | 2.452633815 |
| 8 | 1.74 | 14.96 | 50.82 |  | 8 | 4.827586207 | 4.552139037 | 4.544273908 |
| 16 | 1.34 | 8.89 | 36.48 |  | 16 | 6.268656716 | 7.660292463 | 6.330592105 |
| 24 | 0.8 | 7.75 | 19.54 |  | 24 | 10.5 | 8.787096774 | 11.81883316 |
| 32 | 0.66 | 4.93 | 61 |  | 32 | 12.72727273 | 13.81338742 | 3.785901639 |
| **1c** | Runtime in seconds |  |  |  | Speedup |  |  |  |
| Threads | 960x960 | 1920x1920 | 2880x2880 |  | Threads | 960x960 | 1920x1920 | 2880x2880 |
| 1 | 10.12 | 81.8 | 276.54 |  | 1 | 1 | 1 | 1 |
| 2 | 19.67 | 159.72 | 546.01 |  | 2 | 0.51448907 | 0.512146256 | 0.50647424 |
| 4 | 19.66 | 158.59 | 534.09 |  | 4 | 0.514750763 | 0.515795447 | 0.517777903 |
| 8 | 19.72 | 158.96 | 534.65 |  | 8 | 0.513184584 | 0.514594867 | 0.517235575 |
| 16 | 10.7 | 83.51 | 275.45 |  | 16 | 0.945794393 | 0.97952341 | 1.003957161 |
| 24 | 7.03 | 53.73 | 180.78 |  | 24 | 1.439544808 | 1.52242695 | 1.529704613 |
| 32 | 5.27 | 42.07 | 136.91 |  | 32 | 1.920303605 | 1.944378417 | 2.019867066 |
| **1d** | Runtime in seconds |  |  |  | Speedup |  |  |  |
| Threads | 960x960 | 1920x1920 | 2880x2880 |  | Threads | 960x960 | 1920x1920 | 2880x2880 |
| 1 | 10.13 | 81.93 | 276.52 |  | 1 | 1 | 1 | 1 |
| 2 | 19.52 | 156.87 | 540.67 |  | 2 | 0.518954918 | 0.522279595 | 0.51143951 |
| 4 | 19.86 | 155.6 | 523.41 |  | 4 | 0.510070493 | 0.526542416 | 0.528304771 |
| 8 | 19.77 | 158.41 | 533.12 |  | 8 | 0.512392514 | 0.517202197 | 0.518682473 |
| 16 | 10.4 | 83.05 | 280.52 |  | 16 | 0.974038462 | 0.986514148 | 0.985740767 |
| 24 | 6.68 | 55.16 | 181.96 |  | 24 | 1.516467066 | 1.485315446 | 1.519674654 |
| 32 | 5.15 | 40.86 | 138.05 |  | 32 | 1.966990291 | 2.005139501 | 2.003042376 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1a |  |  |  |  | Num flop (n^3) | Gflops/thread | Num flop (n^3) | Gflops/thread | Num flop (n^3) | Gflops/thread |
| Threads | Runtime in seconds | |  |  | Columns/node: 480 |  | Columns/node: 960 |  | Columns/node: 1920 |  |
| 1 | 1.05 | 8.37 | 67.8 |  | 110592000 | 105.3257143 | 884736000 | 105.7032258 | 7077888000 | 104.3936283 |
| 2 | 0.55 | 4.58 | 36.12 |  | 110592000 | 100.5381818 | 884736000 | 96.58689956 | 7077888000 | 97.97740864 |
| 4 | 0.29 | 2.34 | 18.81 |  | 110592000 | 95.33793103 | 884736000 | 94.52307692 | 7077888000 | 94.0708134 |
| 8 | 0.15 | 1.16 | 9.26 |  | 110592000 | 92.16 | 884736000 | 95.33793103 | 7077888000 | 95.54384449 |
| 16 | 1.06 | 8.11 | 27.36 |  | 884736000 | 52.16603774 | 7077888000 | 54.5459926 | 23887872000 | 54.56842105 |
| 24 | 6.8 | 18.51 | 42.94 |  | 7077888000 | 43.36941176 | 23887872000 | 53.77244733 | 56623104000 | 54.9440149 |
| 32 | 14.84 | 32.37 | 260.08 |  | 23887872000 | 50.30296496 | 56623104000 | 54.6639481 | 4.52985E+11 | 54.42854506 |
| 1b |  |  |  |  | Num flop (n^3) |  | Num flop (n^3) |  | Num flop (n^3) |  |
| Threads | Runtime in seconds | |  |  | Columns/node: 480 |  | Columns/node: 960 |  | Columns/node: 1920 |  |
| 1 | 1.06 | 8.4 | 68.09 |  | 110592000 | 104.3320755 | 884736000 | 105.3257143 | 7077888000 | 103.9490087 |
| 2 | 0.55 | 4.5 | 36.27 |  | 110592000 | 100.5381818 | 884736000 | 98.304 | 7077888000 | 97.57220844 |
| 4 | 0.37 | 2.89 | 27.73 |  | 110592000 | 74.72432432 | 884736000 | 76.53425606 | 7077888000 | 63.81074648 |
| 8 | 0.21 | 1.87 | 14.92 |  | 110592000 | 65.82857143 | 884736000 | 59.14010695 | 7077888000 | 59.29865952 |
| 16 | 1.27 | 8.96 | 37.4 |  | 884736000 | 43.54015748 | 7077888000 | 49.37142857 | 23887872000 | 39.91957219 |
| 24 | 6.62 | 20.42 | 53.22 |  | 7077888000 | 44.54864048 | 23887872000 | 48.74280118 | 56623104000 | 44.33100338 |
| 32 | 16.54 | 36.97 | 380.39 |  | 23887872000 | 45.13276904 | 56623104000 | 47.8623749 | 4.52985E+11 | 37.21384894 |
| 1c |  |  |  |  | Num flop (n^3) |  | Num flop (n^3) |  | Num flop (n^3) |  |
| Threads | Runtime in seconds | |  |  | Columns/node: 480 |  | Columns/node: 960 |  | Columns/node: 1920 |  |
| 1 | 1.27 | 10.12 | 81.88 |  | 110592000 | 87.08031496 | 884736000 | 87.42450593 | 7077888000 | 86.44220811 |
| 2 | 2.49 | 19.77 | 158.77 |  | 110592000 | 22.20722892 | 884736000 | 22.37572079 | 7077888000 | 22.28975247 |
| 4 | 2.47 | 19.77 | 160.11 |  | 110592000 | 11.19352227 | 884736000 | 11.18786039 | 7077888000 | 11.05160202 |
| 8 | 2.45 | 19.66 | 158.7 |  | 110592000 | 5.64244898 | 884736000 | 5.625228891 | 7077888000 | 5.57489603 |
| 16 | 10.2 | 81.63 | 279.18 |  | 884736000 | 5.421176471 | 7077888000 | 5.419184123 | 23887872000 | 5.347775629 |
| 24 | 54.73 | 185.02 | 435.6 |  | 7077888000 | 5.388488946 | 23887872000 | 5.379569776 | 56623104000 | 5.416198347 |
| 32 | 138.15 | 328.57 | 2632.54 |  | 23887872000 | 5.403517915 | 56623104000 | 5.38537298 | 4.52985E+11 | 5.377231115 |
| 1d |  |  |  |  | Num flop (n^3) |  | Num flop (n^3) |  | Num flop (n^3) |  |
| Threads | Runtime in seconds | |  |  | Columns/node: 480 |  | Columns/node: 960 |  | Columns/node: 1920 |  |
| 1 | 1.27 | 10.11 | 81.89 |  | 110592000 | 87.08031496 | 884736000 | 87.51097923 | 7077888000 | 86.43165222 |
| 2 | 2.45 | 19.49 | 156.43 |  | 110592000 | 22.56979592 | 884736000 | 22.69717804 | 7077888000 | 22.6231797 |
| 4 | 2.49 | 19.79 | 157.27 |  | 110592000 | 11.10361446 | 884736000 | 11.17655382 | 7077888000 | 11.25117314 |
| 8 | 2.56 | 19.76 | 158.71 |  | 110592000 | 5.4 | 884736000 | 5.596761134 | 7077888000 | 5.574544767 |
| 16 | 10.58 | 82.8 | 273.91 |  | 884736000 | 5.226465028 | 7077888000 | 5.342608696 | 23887872000 | 5.450666277 |
| 24 | 54.51 | 185.07 | 432.1 |  | 7077888000 | 5.410236654 | 23887872000 | 5.378116388 | 56623104000 | 5.460069428 |
| 32 | 138.26 | 326.62 | 2651.39 |  | 23887872000 | 5.399218863 | 56623104000 | 5.417524953 | 4.52985E+11 | 5.339001807 |

**Appendix B:**