MPI optimizations: synchronization and communication overhead, tips & tricks.

Ahmed Taleb Bechir

I Generalities

1. Give the definition of network latency.
Network latency is the amount of time it takes for data to go across a network from a src to a dst

2. Give the definition of network bandwidth.
Network bandwidth is the rate of data or the capacity of data to be transmitted on a network

3. Give their respective units.
Latency is usually given in milliseconds, while bandwidth is given in bits per second

4. Give the name and version of the MPI library you are using. Give the command used to retrieve this information.
The version of MPI that I'm using is mpich 3.4.1, and we could find out the version using this command:

mpiexec - - version
or mpirun - - version

5. To make sure your installation works, you can use the hostname command, which returns the name of the current process's host:

mpirun -np 2 hostname

What happens when all processes are executed on the same host?
Each process will return the same hostname, MPI handles the communications between the different processes to make sure the program is running correctly

II Ping Pong

6.
Using OpenMPI 2.0.1, we measured the execution time as a function of message size. We obtain 53.1654 ms for a size of 4,000 Bytes, and 2.0006 s for a size of 4,096 Bytes. How do you explain this difference? What did we actually measure?

This difference is due to the sizes of MPI buffer communication, which is due to the MPI implementation. If the message size exceeds the buffer, then the overhead for the communication increases dramatically
7.Find the message size for which this gap appears in your MPI implementation. This size might be different than the one we observed using OpenMPI 2.0.1 (4,096 B). Explain the reason why.

For my MPI implementation, the size is 2^16 bytes due to differences in MPI implementations. MPICH and OpenMPI may have different optimizations, communication strategies, and buffer sizes

8. We decide to remove the call to sleep() . Does the measure make sense now? Why?

No because the time it takes to actually send the message is negligible and we won't be able to effectively see the diffrences

9.see code

10. Write an actual MPI ping-pong (rank 1 answers back to rank 0). Explain why the measured times corresponds to a message exchange.

The measured times correspond to a message exchange because each process starts measuring time just before initiating its send operation and stops measuring time after completing its receive operation. Therefore, the measured time includes the time taken for both the send and receive operations to complete, as well as any communication overhead involved in transferring the message between the processes.

II.2 First measures

11. Do multiple measures with multiples of 32 Bytes for the message sizes. What do you see?
We can see variations in the execution time.

12. We propose adding a barrier before the message exchange phase. Explain what is the interest of this barrier with regards to the accuracy of the measurements.
its to get a baseline of a time measurement.

13. Rerun the measures of question 11. Are the changes from the previous question sufficient? Why?
Yes they are , as because of the barrier the different process start at the same time and thus we can minimize the different noise that might have influenced the measurements .

II.3 Repetitions

When taking a measurement, it is important to remember that our environment does not allow us to reproduce the exact conditions between each run. What is more, time

measurement itself is fraught with uncertainty. uncertainty. To solve this problem, we prefer to repeat a measurement and calculate an average (mean) and/or a median.

14.see code

II.4 Impact of message size

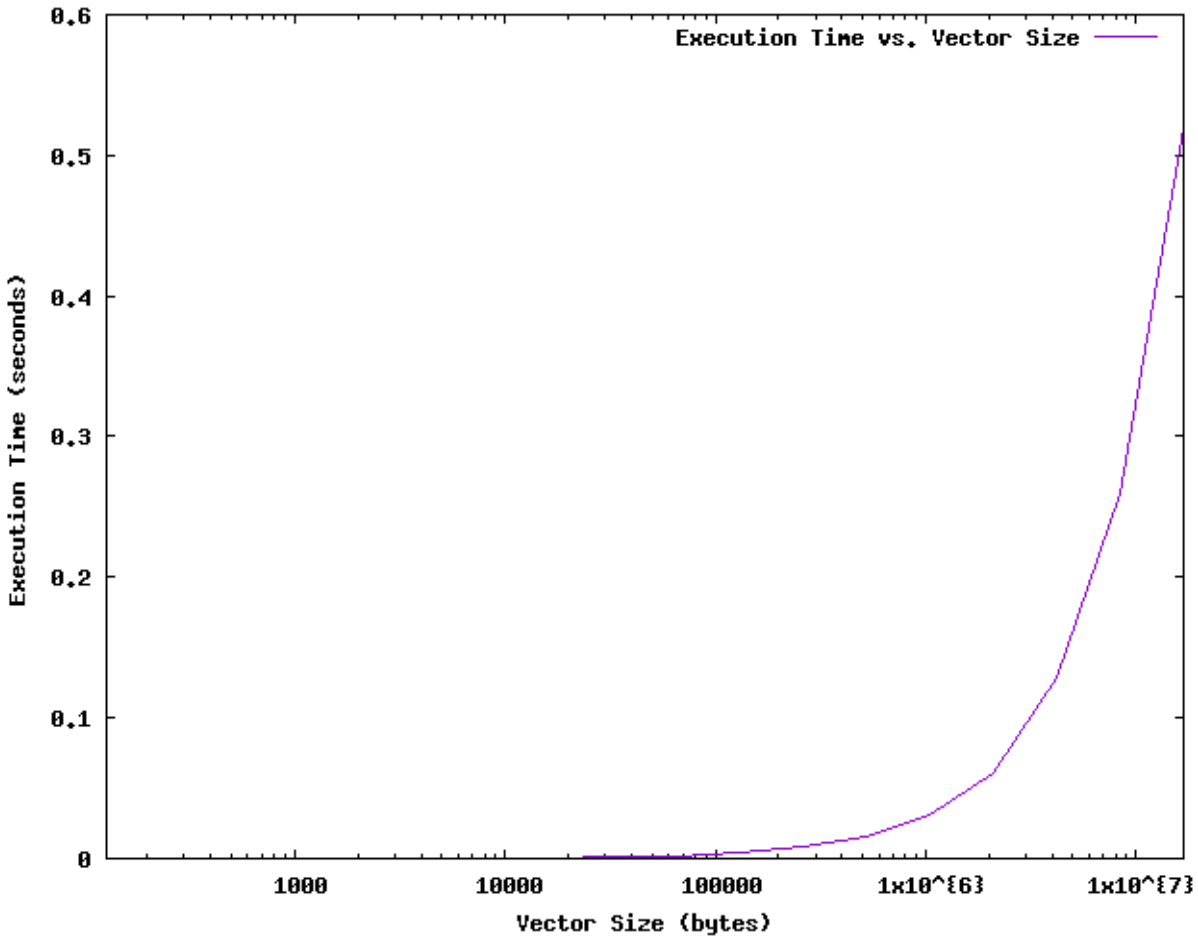
15. Explain why there is such a big gap between local and Infiniband for small sizes. Explain this gap shrinks progressively once the message size increases.

it's due to latency, the delay is more noticeable for small messages because the network has a bigger overhead. So, even though Infiniband is fast, smaller messages are going to end up slower compared to the transmission time.

16. From the previous results, should we send distinct sets of data separately? Or should we try to group them together? Is it true for all sizes?
By grouping data sets together in packets we might be able to reduce that overhead but it works on smaller sizes of data

III Experimental evaluation of scalability

17. Implement this benchmark like so:



18. What does strong scalability represent?

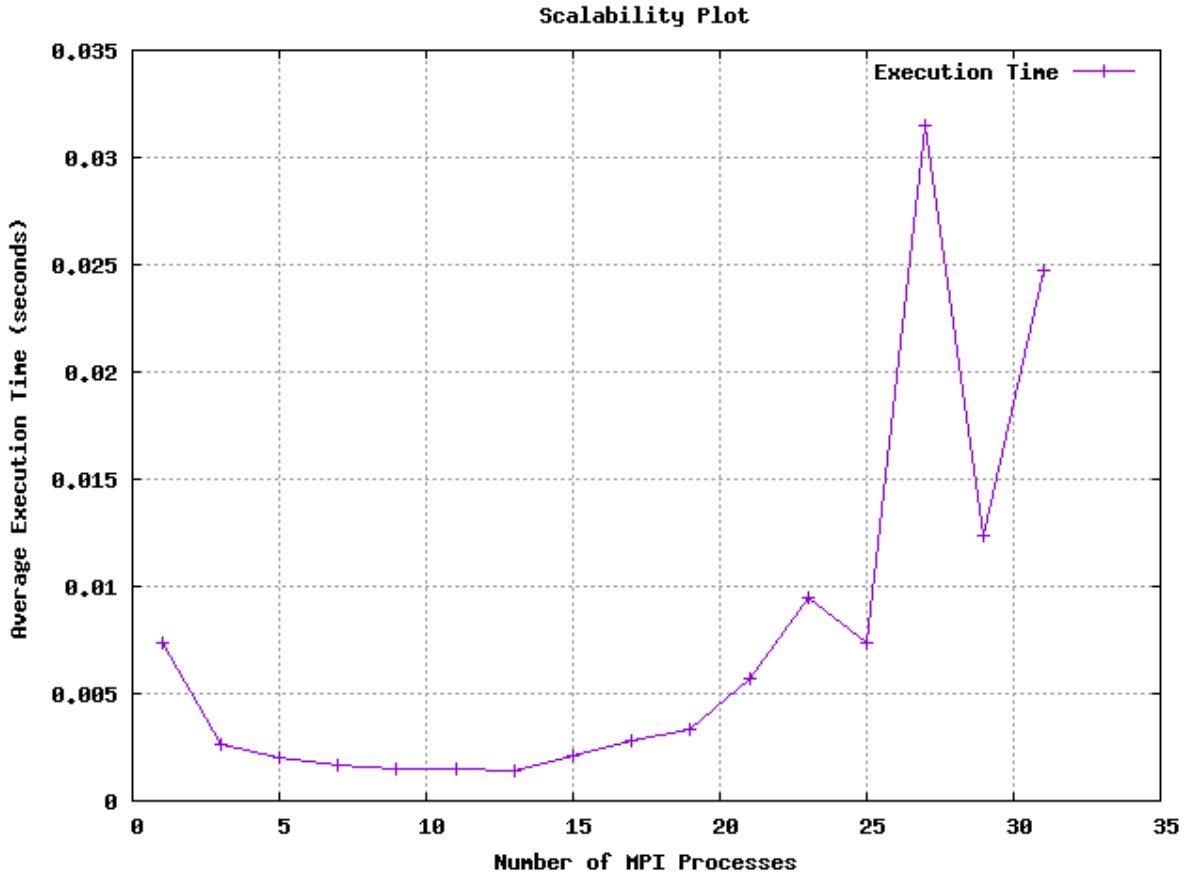
strong scalability is Maintaining performance as resources increase for a fixed problem size.

19. What does weak scalability represent?

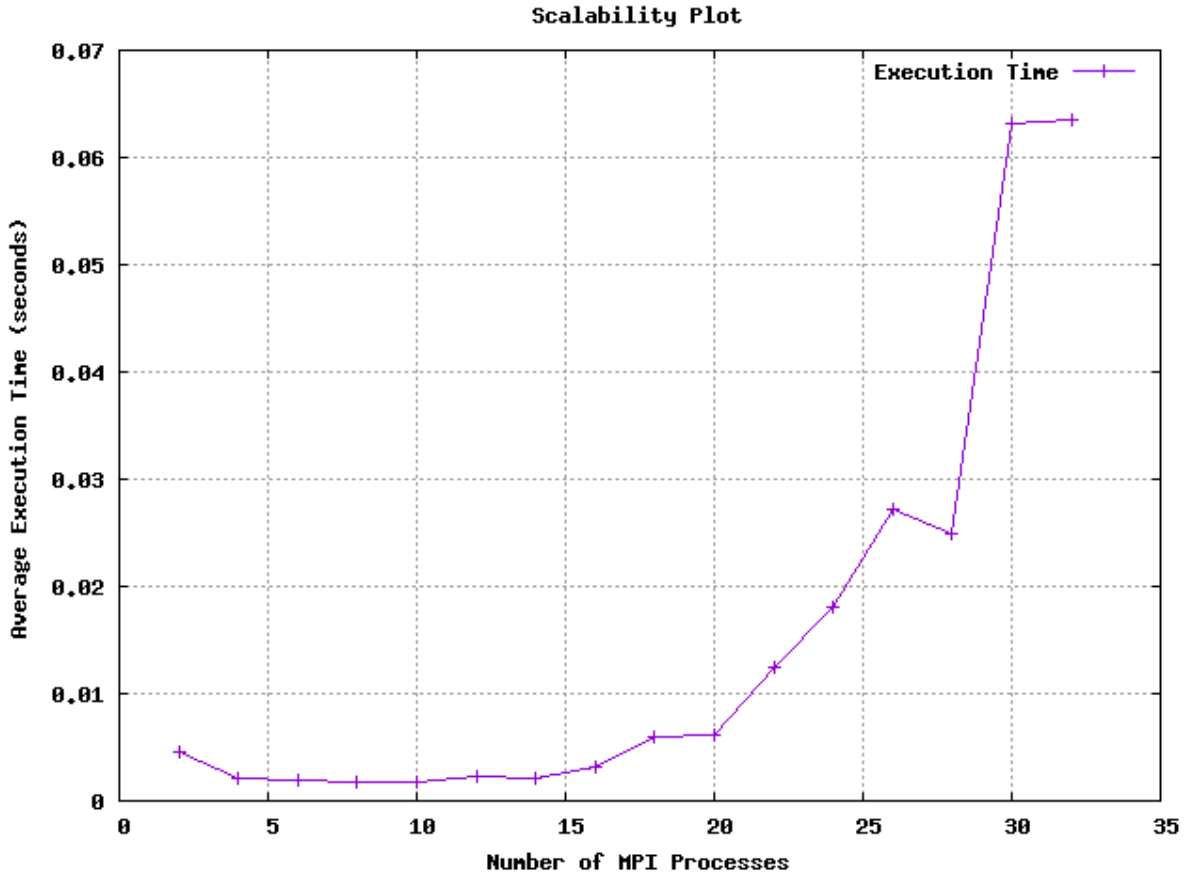
weak scalability is maintaning performance as both problem size and resources increase proportionally.

20. To introduce communications in our program, we want to consider a case inspired by finite elements method (FEM). Our equation is now:

$$X\,(t+1,\, i) = (X(t,\, i-1) + 2X(t,\, i) + X(t,\, i+1))/4$$



21. Introduce a global synchronization in the repetitions loop and reevalute the application's scalability. What do you see? What remarks can you make about communications and the use of barriers in
MPI applications? What should you do to avoid this problem?



we can see clear signs of weak scaling in our plot . we can see that the overhead associated with coordinating a larger number of processes outweighs any potential benefits from parallelism.

IV Collectives and algorithms in OpenMPI

22. Find what are the usable algorithms for the MPI_Bcast routine. Compare their performance depending on the number of MPI processes and buffer size.

1 basic linear,
2 chain,
3:pipeline,
4: split binary tree,
5: binary tree,
6: binomial tree,
7: knomial tree,
8: scatter_allgather,
9: scatter_allgather_ring