Saint Petersburg Electrotechnical University "LETI"

ANALYSIS OF THREAD SYNCHRONIZATION EFFICIENCY IN HYBRID MPI+THREADS PARALLEL PROGRAMS

Student: Qian Xinyu

Supervisor: Alexey Paznikov

May 26, 2023

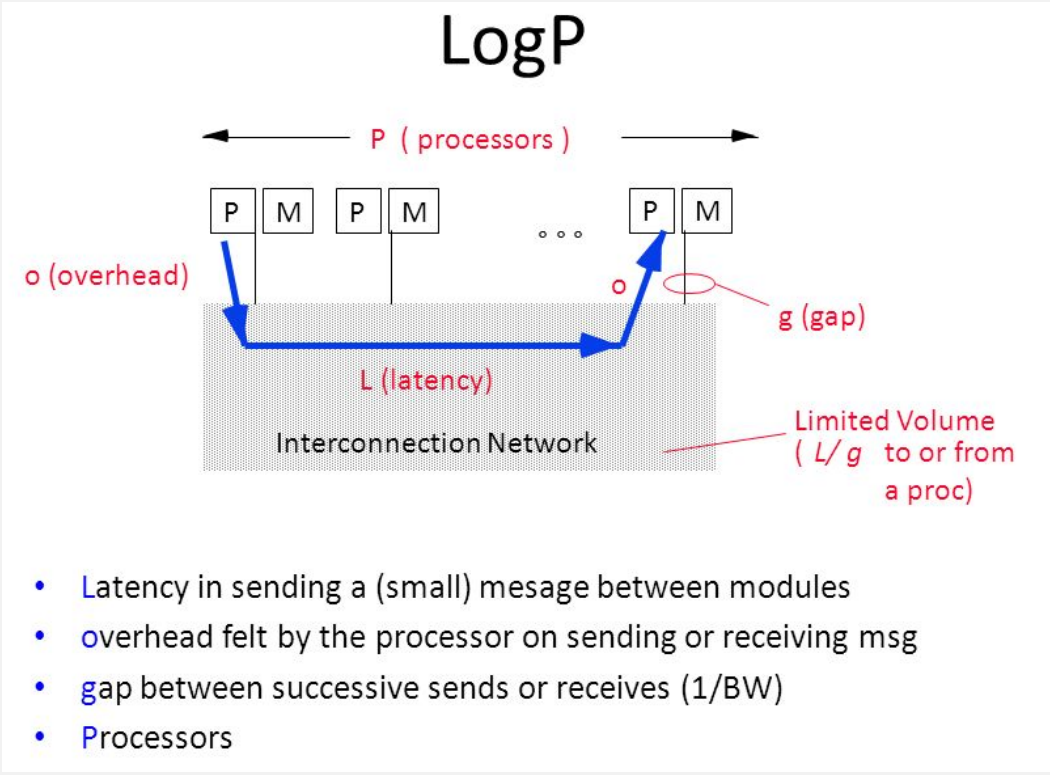
Introduction

Research status:

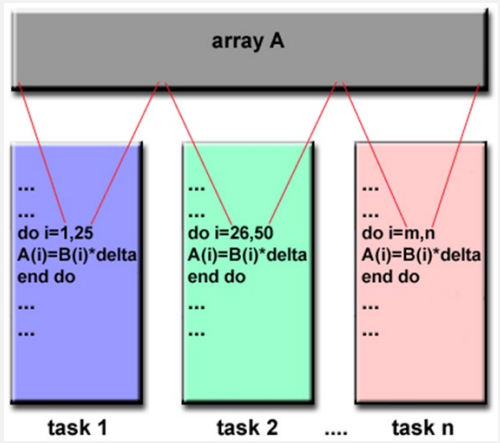
- The importance of high-performance computing (HPC) in modern scientific research and engineering computing
- Classic Parallel Computing Models(LOGP)
- Wide application of MPI programming model
- Disadvantages of the pure MPI model

Research objectives:

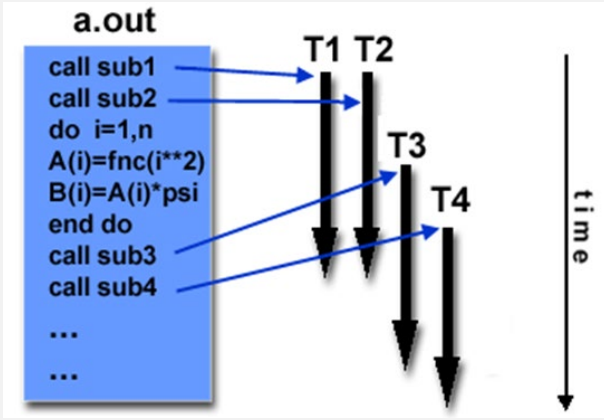
- Implement hybrid MPI+threads parallel program
- Evaluate the thread synchronization efficiency
- Optimize parallel communication and computation processes



Research methods - Parallel theory

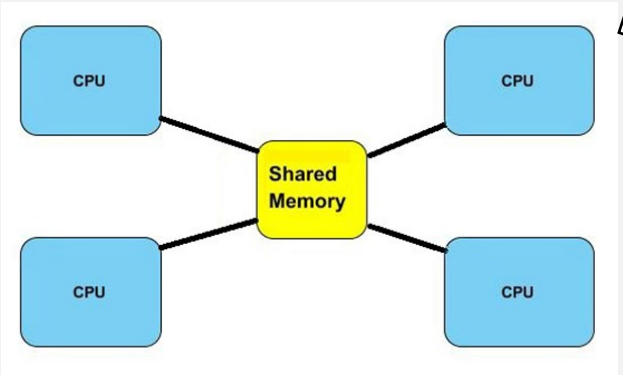


Data Parallelism Model

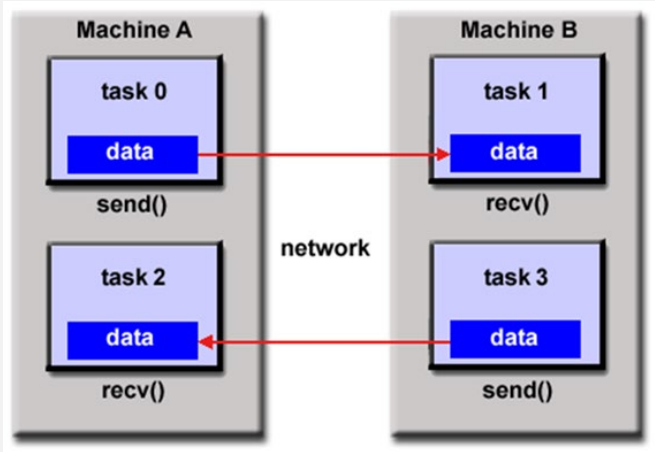


Thread-level parallelism

NSU cluster
MIMD
SMP
Distributed Memory



Shared Memory Model



Message Passing Model

MPI communication

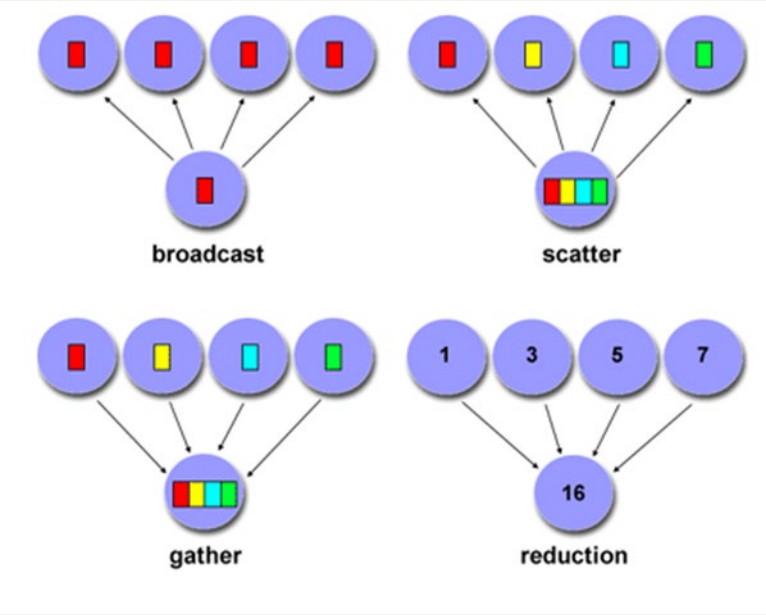
Point-to-point function:

- MPI_Send( )
- MPI_Recv( )
- MPI_Isend( )
- MPI_Irecv( )

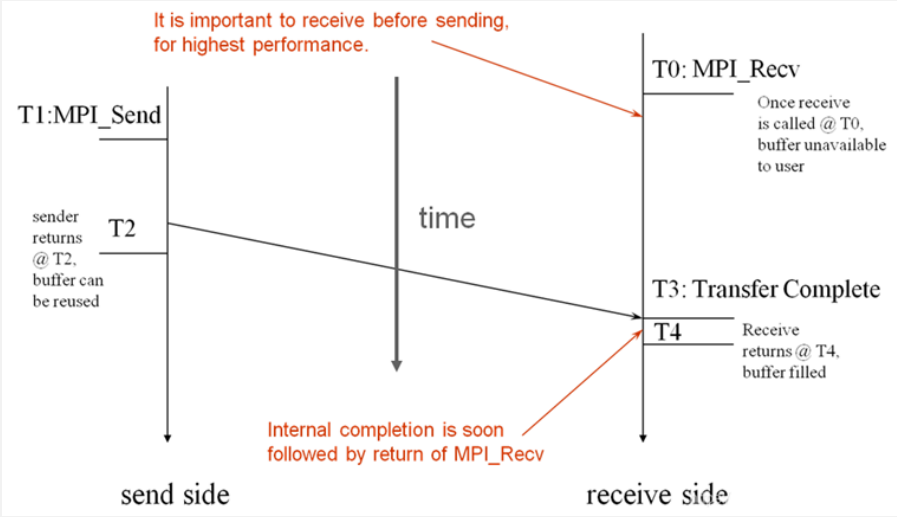
Collective function:

- MPI_Bcast( )
- MPI_Scatter( )
- MPI_Gather( )
- MPI_Reduce( )

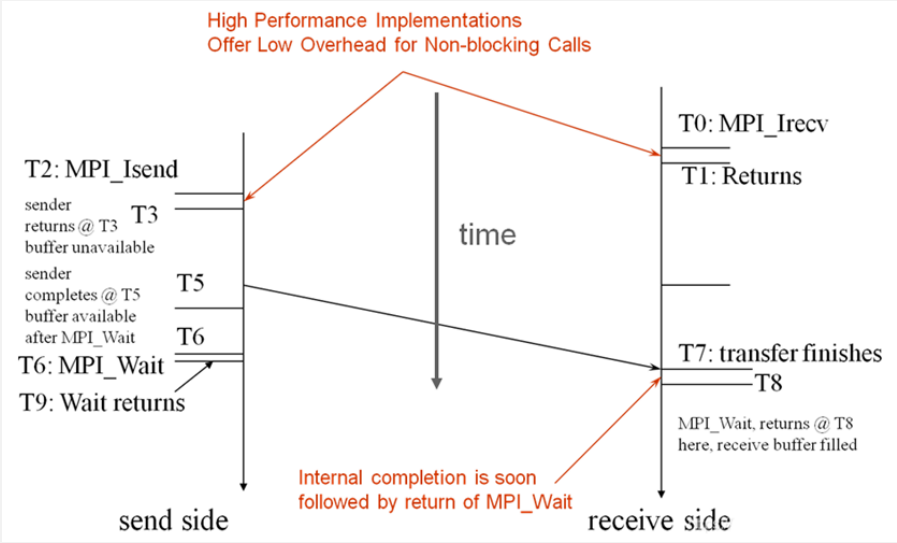
MPI collective Communication



MPI blocking send and receive



MPI non-blocking send and receive



Research methods - Performance Metrics

Amdahl's law

$$S = \frac{1}{1 - a + \frac{a}{n}}$$

S - speedup
a - the proportion of parallel computing
n - the number of parallel nodes processed

Gustafson's Law

$$\text{Acceleration ratio (S)} = P + (1\text{-}P) \times N$$

P - the proportion of the parallel part of the program to the total execution time
N - the number of processors

Communication-to-Computation Ratio

$$CCR = \frac{communication\ time}{calculation\ time}$$

communication time - the time required to transmit data between processors

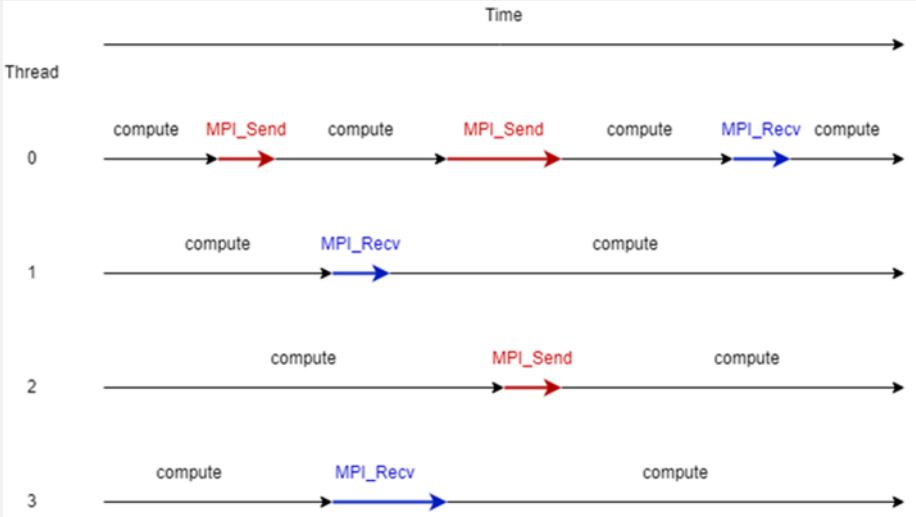
Measuring Time with MPI:

```
start = MPI_Wtime();
finish = MPI_Wtime();
printf(" time: %lf s \n", finish - start);
```

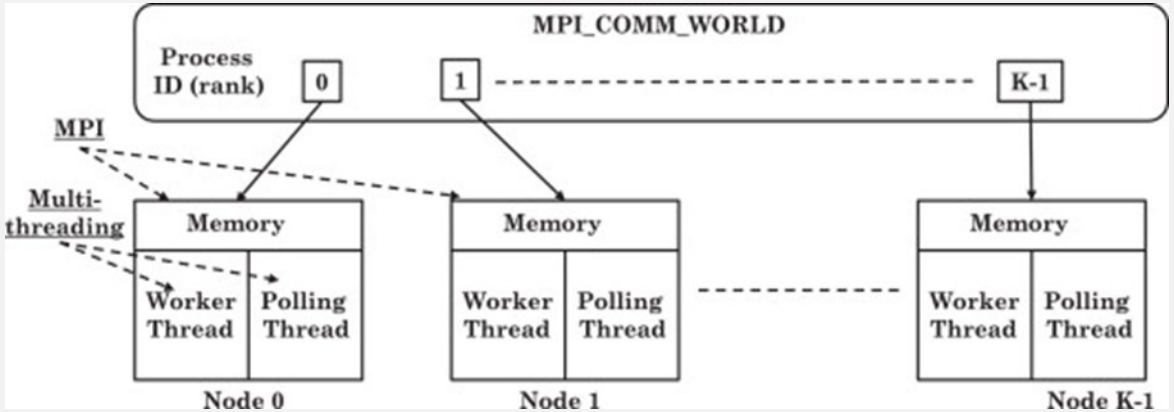
Overview of the hybrid model

MPI_THREAD_MULTIPLE — any thread in each process can call MPI functions at the same time without restrictions.

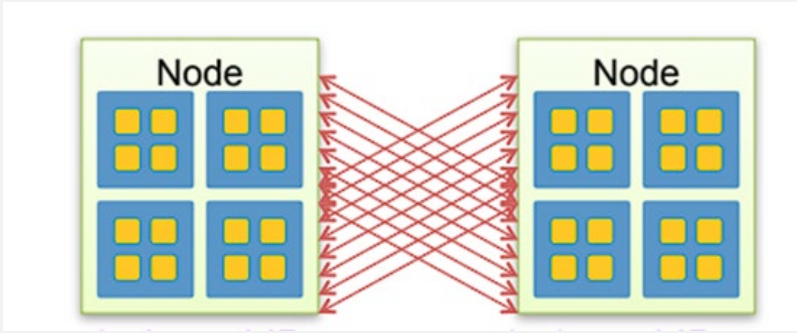
- MPI messaging communication between nodes
- OpenMP or Pthreads shared memory in a single node
- Multithreaded concurrent MPI communication
- MPI deadlock situation and avoidance method



MPI communication under multithreading



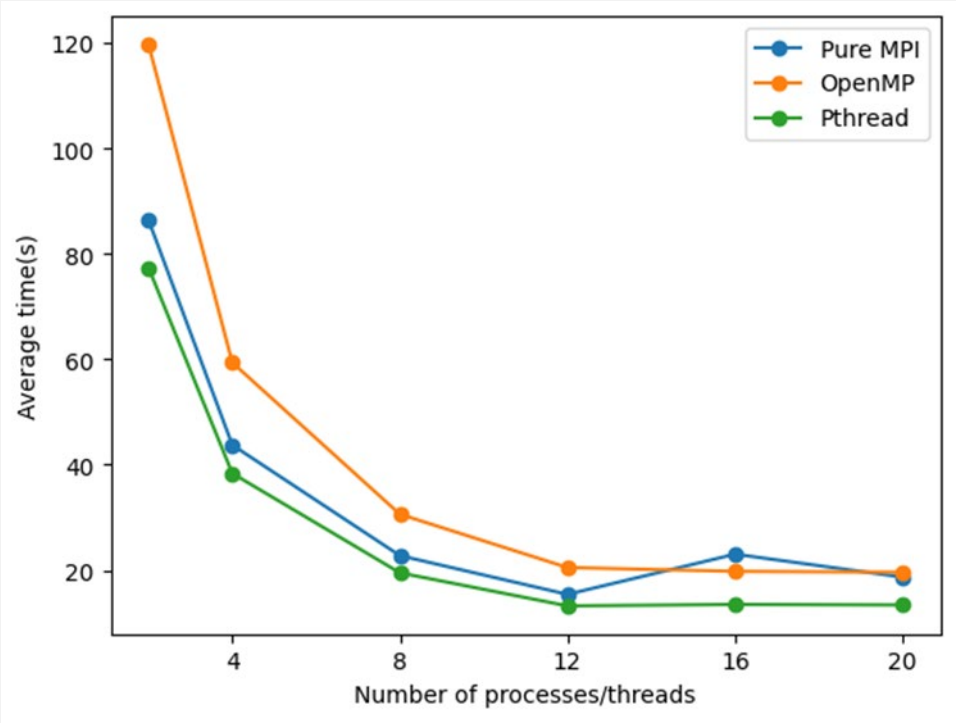
Hybrid MPI model



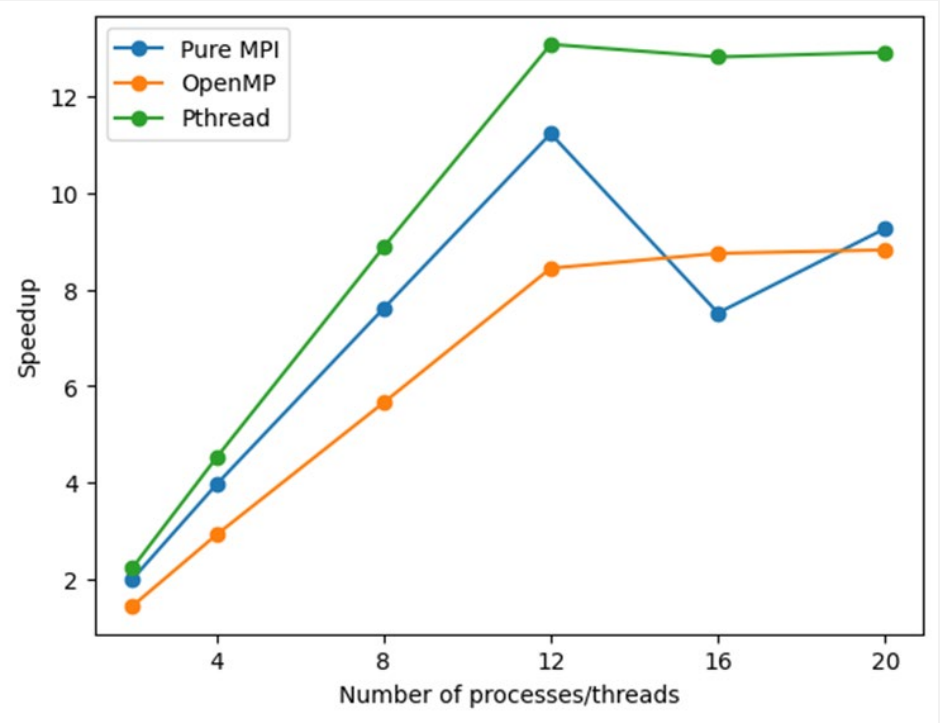
Rank-thread ID to any Rank-thread ID

Process performance vs Thread performance

Time(s)



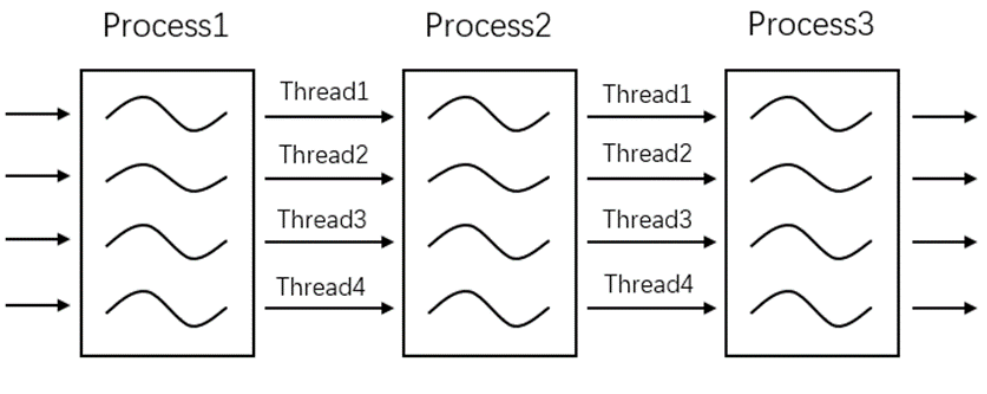
Speedup ratio



- MPI process has good performance at coarse-grained levels (better than OpenMP).
- The performance of thread synchronization is good at fine-grained.
- Pthreads has the best performance.

Multithreaded ring communication

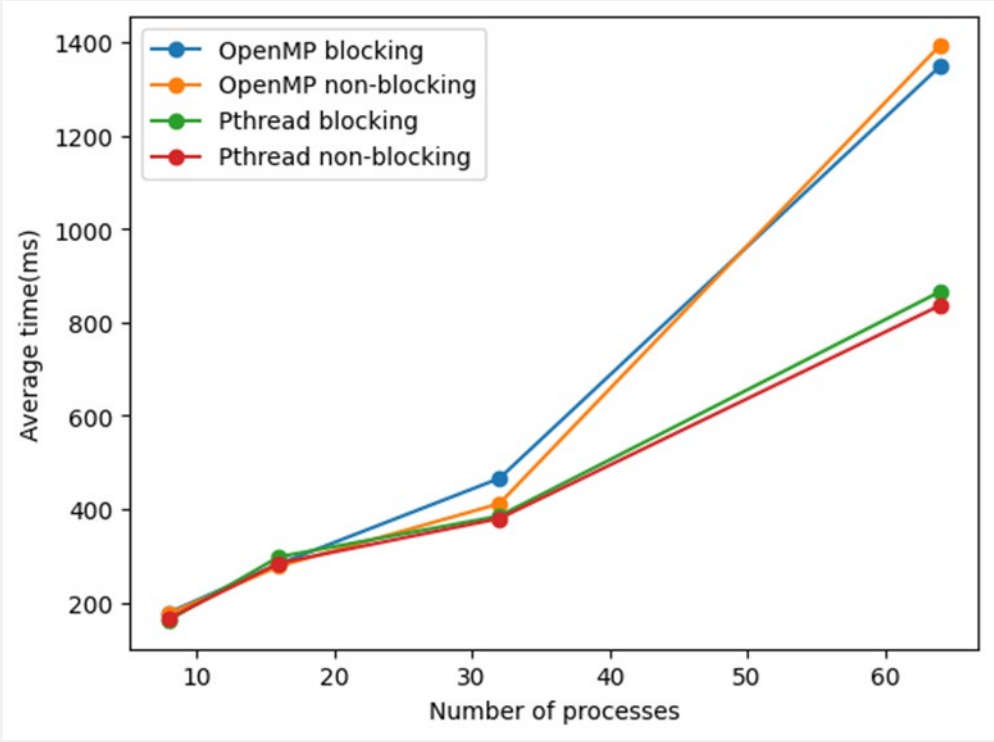
Ring communication model




```
#pragma omp parallel
  {
      int thread_id = omp_get_thread_num();
      int num_threads = omp_get_num_threads();
      int send_buf = rank * num_threads + thread_id;
      int recv_buf;
      int source = (rank - 1 + size) % size;
      int dest = (rank + 1) % size;
      MPI_Request send_request, recv_request;
      MPI_Isend(&send_buf, 1, MPI_INT, dest, thread_id, MPI_COMM_WORLD, &send_request);
      MPI_Irecv(&recv_buf, 1, MPI_INT, source, thread_id, MPI_COMM_WORLD, &recv_request);
      MPI_Wait(&send_request, MPI_STATUS_IGNORE);
      MPI_Wait(&recv_request, MPI_STATUS_IGNORE);
}
```

In non-blocking mode, we use the MPI_Wait function under each thread to block the thread until the MPI request is completed.

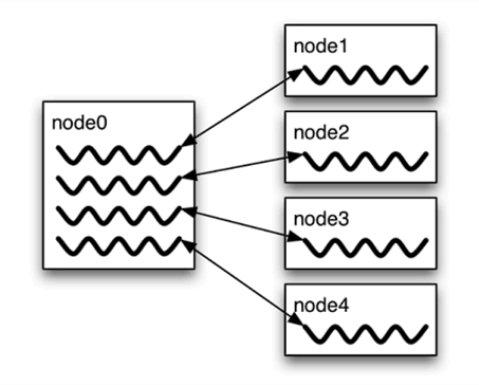
Time(s)



With the growth of the number of processes, that is, the increase of the number of thread communications, Pthreads has more and more obvious synchronization performance advantages compared to OpenMP.

Multithreaded linear broadcasting

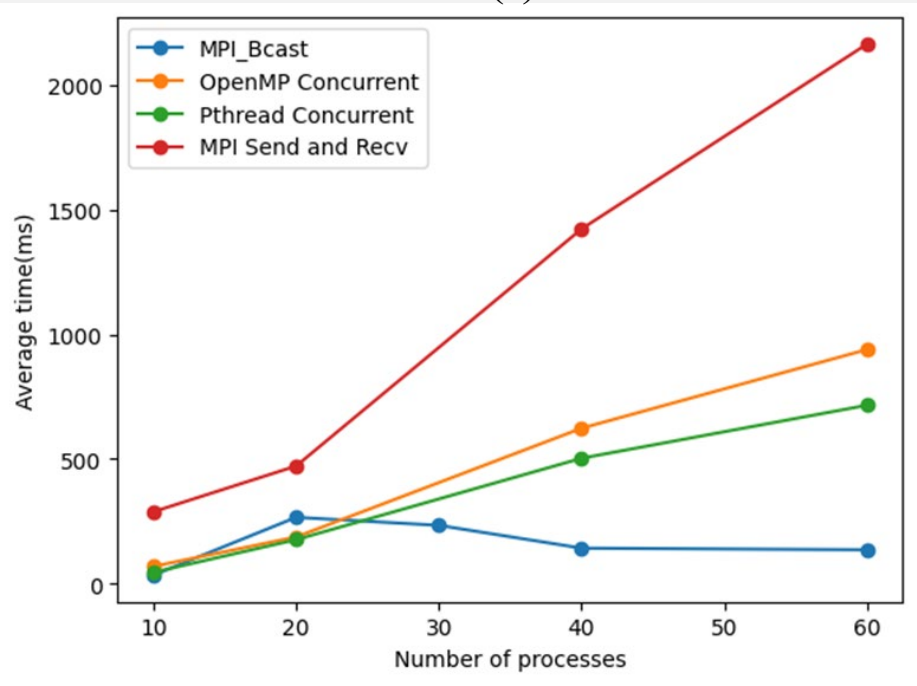
Linear broadcast model



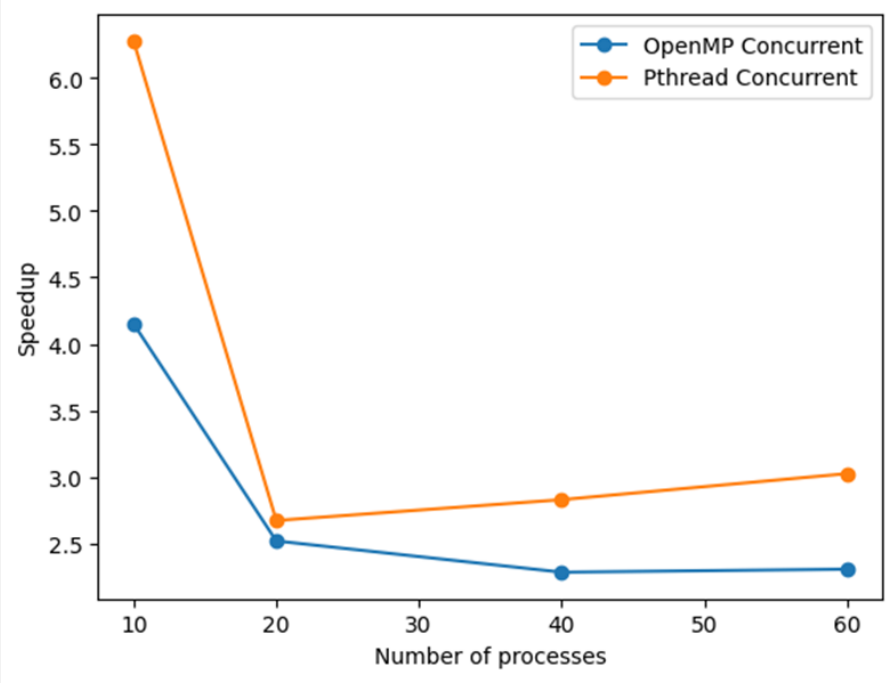
In MPI blocking mode:

- MPI_Bcast can broadcast faster as the number of processes increases.
- Speedup ratio of OpenMP and Pthreads was the largest at the beginning.
- Speedup ratio of OpenMP gradually tends to 2.
- Analysis: Speedup ratio of Pthreads should eventually be closer to 2.

Time(s)



Speedup ratio



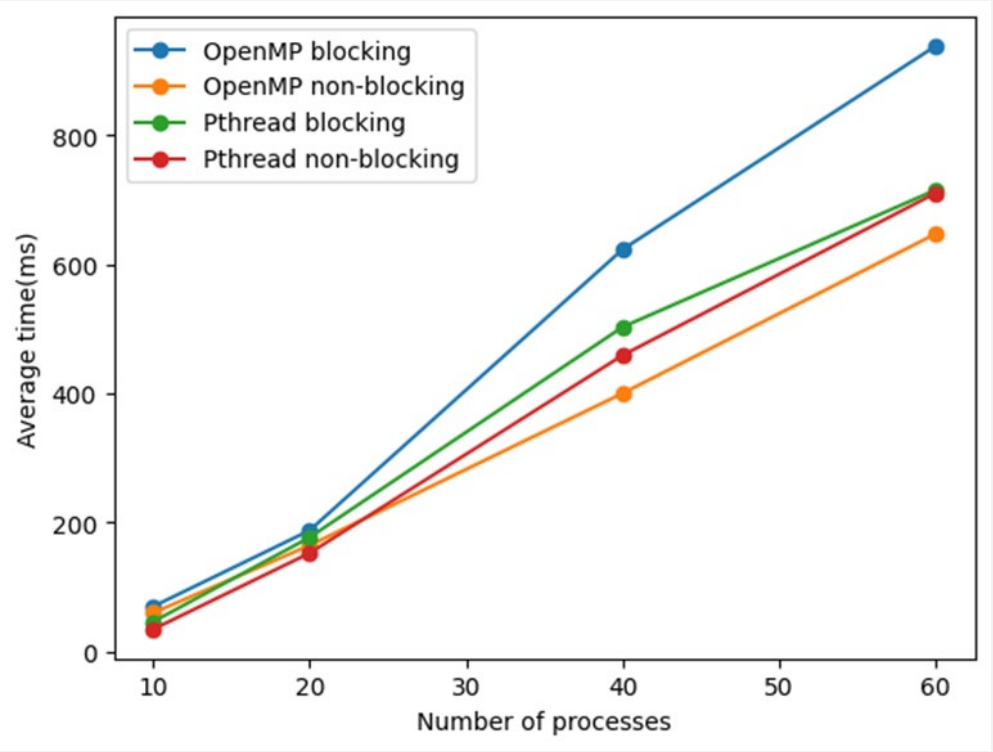
Blocking performance vs non-blocking performance

Code sample:

```
#pragma omp parallel private(i) num_threads(nthread)
{
    #pragma omp for schedule(static)
    for (i = 1; i < size; i++)
    {
        MPI_Isend(value, 3000*3000, MPI_INT, i, 0, MPI_COMM_WORLD,
&request[i-1]);
    }
}
    MPI_Waitall(size - 1, request, statuses);
```

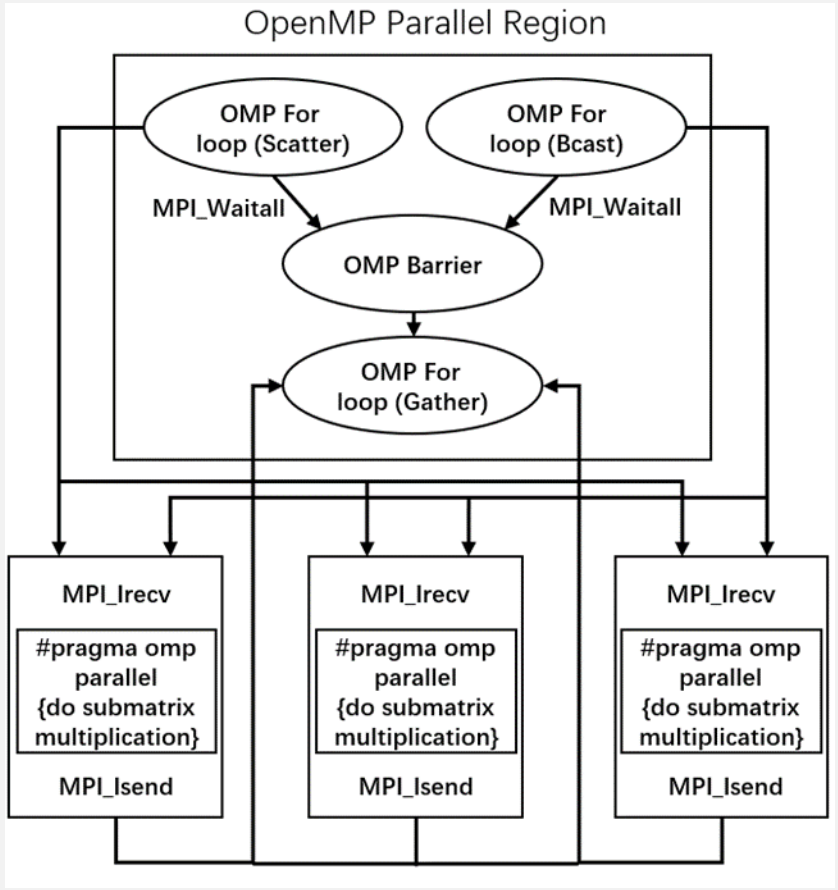
In non-blocking mode, we use the MPI_Waitall function in the main process to block the main thread to ensure that the synchronization of all thread communication is completed.

Linear broadcasting



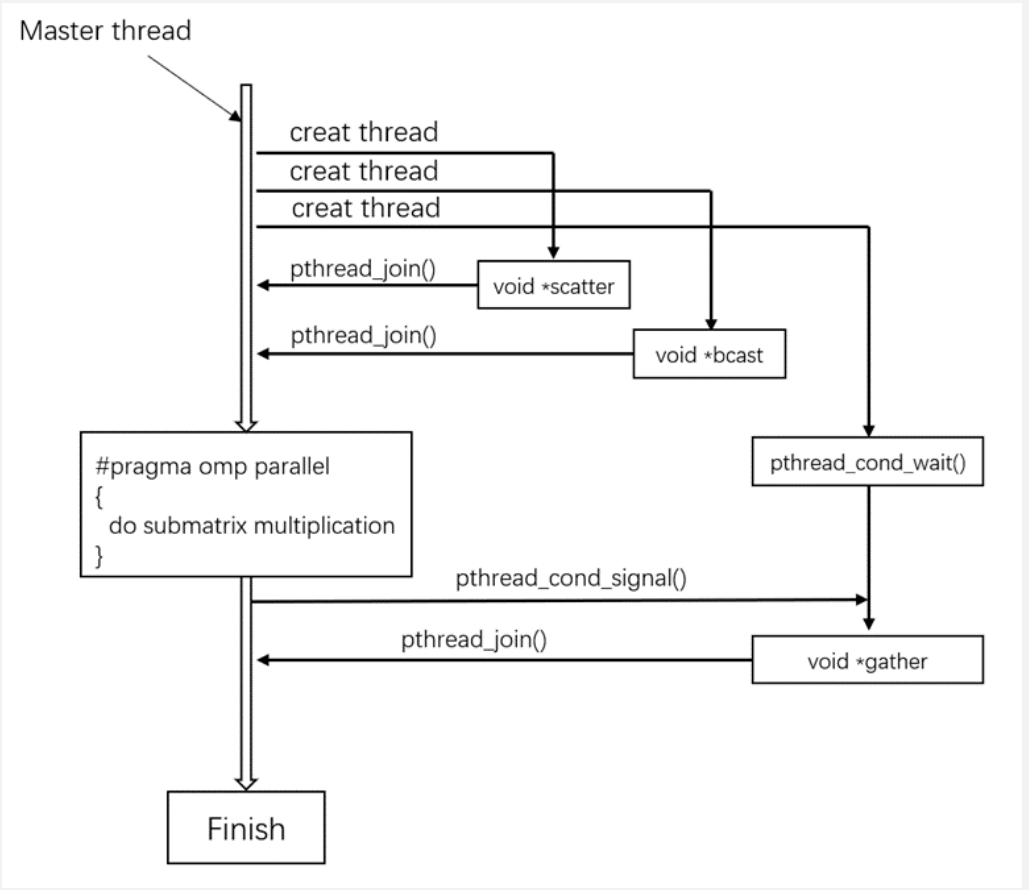
- Under the previous communication model, the performance of MPI blocking and non-blocking is almost indistinguishable.
- Under this linear broadcast model, the performance of multithreaded non-blocking sending and receiving is obviously better than that of multithreaded blocking sending and receiving.

Matrix multiplication with hybrid model



1. hybrid MPI+OpenMP model

'#pragma omp barrier' ensures that all threads responsible for scatter and broadcast complete communication

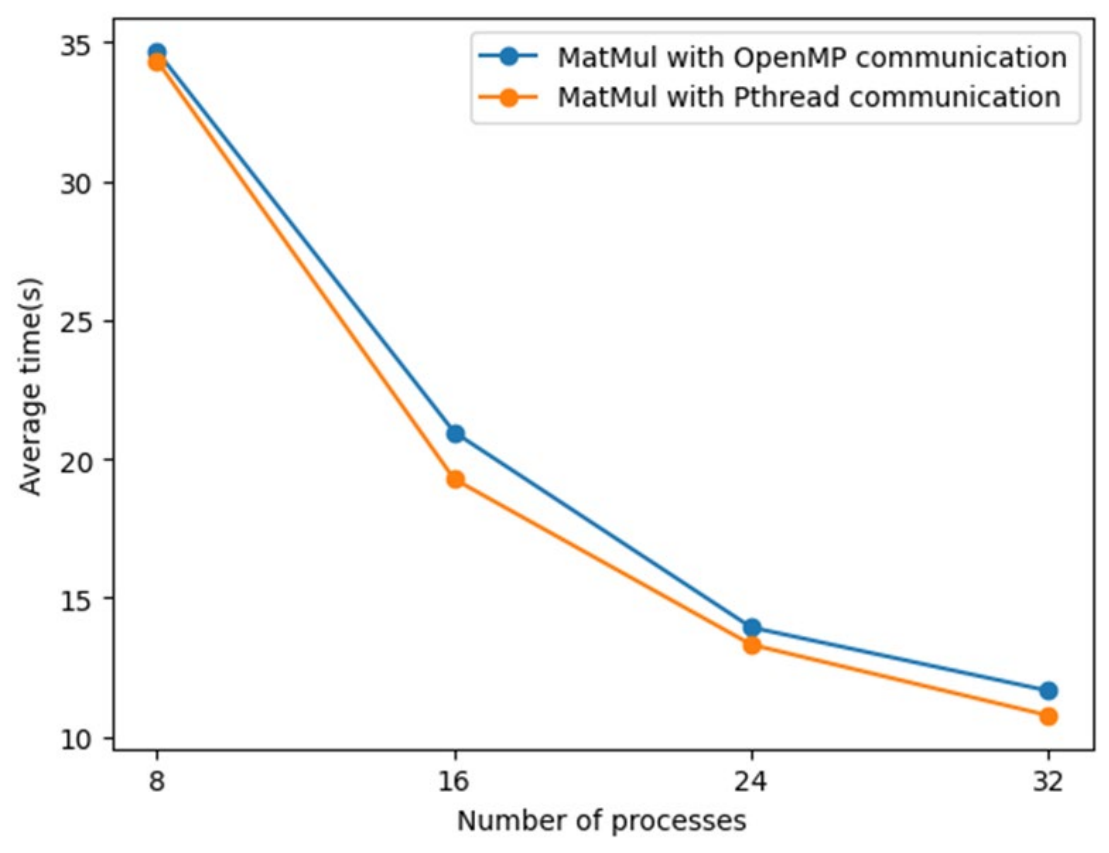


2. hybrid MPI+OpenMP+Pthreads model

'pthread_cond_wait' and 'pthread_cond_signal' are used for synchronization between threads. The thread responsible for collecting waits for the main thread to complete the calculation before being awakened.

Comparison of two matrix multiplication models

Time(s)



- At different parallel granularity, Pthreads still improves the degree of parallelism for communication compared to OpenMP, and further reduces the overhead of communication synchronization.
- Therefore, Pthreads can better reduce the communication-to-calculation ratio of the entire parallel computing program.

Conclusion

- In the single-node cluster experiment, Pthreads showed excellent performance, especially in fine-grained parallel tasks, surpassing MPI processes and OpenMP.

- In the multi-node cluster experiment, we found that multi-threaded MPI communication can significantly improve performance compared to single-threaded MPI communication. The non-blocking mode does not have a significant advantage in ring communication, but it saves a lot of time in the linear broadcast mode.

- In the parallel computing experiment to solve matrix multiplication, although the main program execution time is occupied by calculations and MPI communication accounts for only a small part, Pthreads still improves the parallelism of communication relative to OpenMP and reduces the overhead of communication synchronization.

Acknowledgement

Thank you for listening!

Цянь Синьюй, гр. № 7300
Github: https://github.com/QXY716
тел. +7 9213296746
e-mail. 2387995457@qq.com