FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION
OF HIGHER EDUCATION
ITMO UNIVERSITY

Parallel algorithms for the analysis and synthesis of data on the assignments No.6, 7, 8

Performed by
Ivan Dubinin
J4132c

St. Petersburg
2021

Assignment 6

Compile the example Assignment6.c in detail, run it and explain it.

Transform the program using the MPI_TAG field of the status structure in the condition.

Listing of the program

See it in my github repo

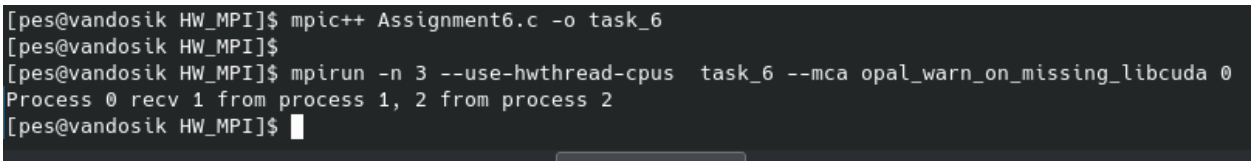
Description

This programm is executed spawning 3 processes.

Process with rank1 sends int value, process with rank 2 – float value. If the 0 proc receives message from rank 1 firstly, it will be decoded, then message from process 2. Else, first will be decoded float message.

This can be also achieved using different tags (e.g. 5 for int, 6 for float). Check tag of the received message, if it is tag for int, decode it first, else decode float message first. Process 0 also receives messages by their TAG, parameter SOURCE in MPI_Recv call are set to MPI_ANY_SOURCE.

Example of launch parameters and output



Assignment 7

Write an MPI program that implements the dot product of two vectors distributed between processes. Two vectors with a size of at least 1,000,000 elements are initialized at process zero and filled with “1”, then they are sent in equal parts to all processes. Parts of vectors are scalar multiplied on each process, the result is sent to he root process and summed up. The total is displayed

Scalar product for two vectors a = [a1, a2, ..., an] and b = [b1, b2, ..., bn] in n-dimensional space defined as:

$$a \cdot b = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n.$$

Listing of the program

See it in my github repo

Description

Program for this task works with n processes (n must be a divider of 1000000). Root process (with 0 rank) creates two arrays of size 1000000. He also counts the number of elements for each part and fillst each part with integers, starting with 1 (Set 1 to first part, 2 to second, etc). Then he sends to other processes number of elements of each vector part to receive(MPI_Bcast), distributes array sequally between all processes (MPI_Scatter). After that every process calculates local sum of dot product of gained parts of two arrays and transfer it to the root process, which aggregate its values in one by summing them(MPI_Reduce). In the end, everyprocess outputs found local dot product, root process also outputs found resulting sum.

Results of the programm execution are shown on the picture below. I have launched it for number of threads equal to 4 and 8.

Example of launch parameters and output

```
[pes@vandosik HW_MPI]$ mpic++ Assignment7.cpp -o task_7
[pes@vandosik HW_MPI]$ mpirun -n 4 --use-hwthread-cpus task_7 --mca opal_warn_on_missing_libcuda 0
Two vectors of 1000000 elements were generated and divided to 4 parts of 250000 elements
Process: 2: local sum = 2250000
Process: 3: local sum = 4000000
Process: 1: local sum = 1000000
Root process: 0: local sum = 250000
Root process: 0 resulting SUM = 7500000
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus task_7 --mca opal_warn_on_missing_libcuda 0
Two vectors of 1000000 elements were generated and divided to 8 parts of 125000 elements
Process: 6: local sum = 6125000
Process: 5: local sum = 4500000
Process: 7: local sum = 8000000
Process: 1: local sum = 500000
Process: 3: local sum = 2000000
Root process: 0: local sum = 125000
Root process: 0 resulting SUM = 25500000
Process: 2: local sum = 1125000
Process: 4: local sum = 3125000
[pes@vandosik HW_MPI]$
```

Assignment 8

Write an MPI program in which two processes exchange messages, measure the timeper exchange iteration, and determine the dependence of the exchange time on themessage length. Determine the latency and maximum achievable bandwidth of thecommunication network. Print the message length in bytes and the throughput inMB/s to the console. Change the length of the message in a loop starting from 1element and increase to 1,000,000 elements, increasing by 10 times at eachiteration.Bandwidth measurement technique:The following technique is used to measure point-to-point bandwidth. Process 0sends a message of length L bytes to process 1.L = number of elements * length in bytes of data typeFor example:L = 100 * sizeof(int);Process 1, having received a message from process 0, sends it a reply message of thesame length. Blocking MPI calls are used (MPI_Send, MPI_Recv). These actions arerepeated N = 10 times in order to minimize the error due to averaging. Process 0measures the time T taken for all these exchanges.The bandwidth R is determined by the formula:

$R=2NL/T$

Latency measurement technique:

Latency is measured as the time it takes to transmit a signal or message of zerolength. At the same time, to reduce the influence of the error and low resolution ofthe system timer, it is important to repeat the operation of sending a signal andreceiving a response,a large number of times. Thus, if the time for N iterations ofsending zero-length messages back and forth was T sec., Then the latency ismeasured as:
$s=t/2N$

Listing of the program

See it in my github repo

Description

Program for this task works with 2processes. Process 0 sends and then receives messages(arrays of integers), process 1 does the opposite: it receives and then sends messages. Message length is initialized with 0 to measure Latency, then it is set to 1 and increases 10 times until it reaches 1,000,000. Each process outputs

information about average time of sending and receiving message with certain length, channel and calculated bandwidth. For messages of 0 len Latency is calculated. To minimize the error due to averaging send-recv sequence was repeated 10 times for each msg size.

As one can see on the picture below the Bandwidth is rising with increasing of msg length.

Example of launch parameters and output