FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION
OF HIGHER EDUCATION
ITMO UNIVERSITY

Parallel algorithms for the analysis and synthesis of data
on the assignments No.9, 10, 11

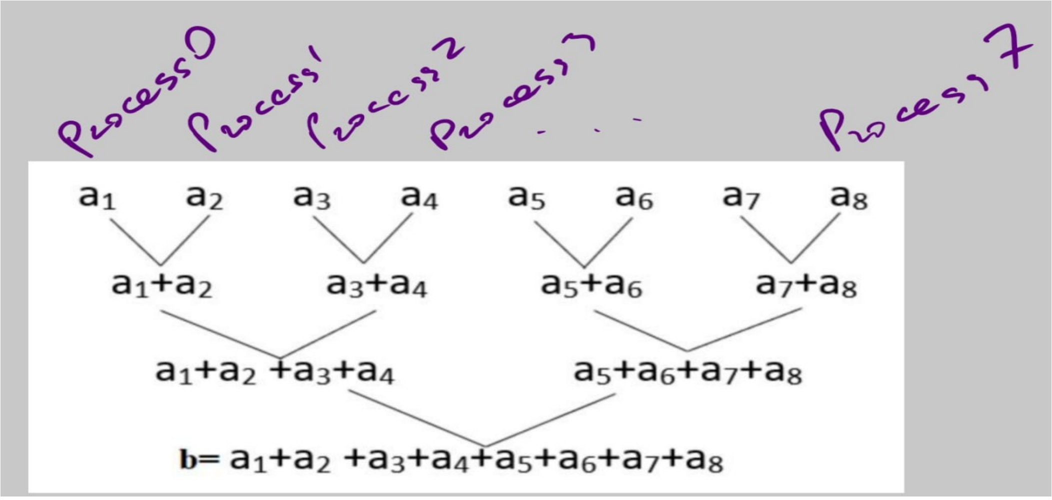
Performed by
Ivan Dubinin
J4132c

St. Petersburg
2021

Assignment 9

Write an MPI program in which the global vector addition operation is modeled by adoubling (cascade) scheme using point-to-point data transfers. Compare the execution timeof such a simulation using the MPI_Reduce procedure on as many processes as possible.Each process stores an array of 1,000,000 elements equal to ‘1’.The resulting array is saved, for example, on process 0. To check the result, print the first 5 elements for both methods to the console.

For implementing cascade scheme the following destribution of data was used. In my case a_i stands for array parts (+ is vector sum in this case)


Process0
Process1
Process2
Process3
. . .
Process7
a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8
a_1+a_2 a_3+a_4 a_5+a_6 a_7+a_8
$a_1+a_2+a_3+a_4$ $a_5+a_6+a_7+a_8$
$b= a_1+a_2+a_3+a_4+a_5+a_6+a_7+a_8$

Listing of the program

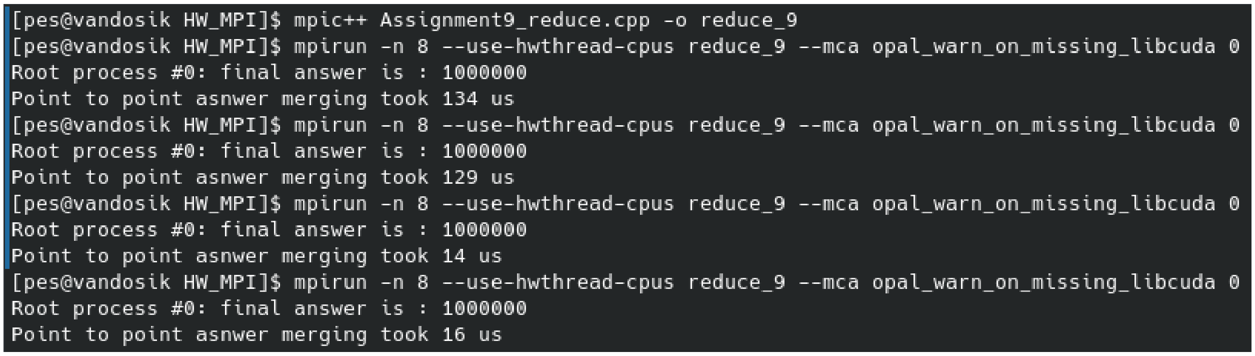
Cascade variant

Reduce variant

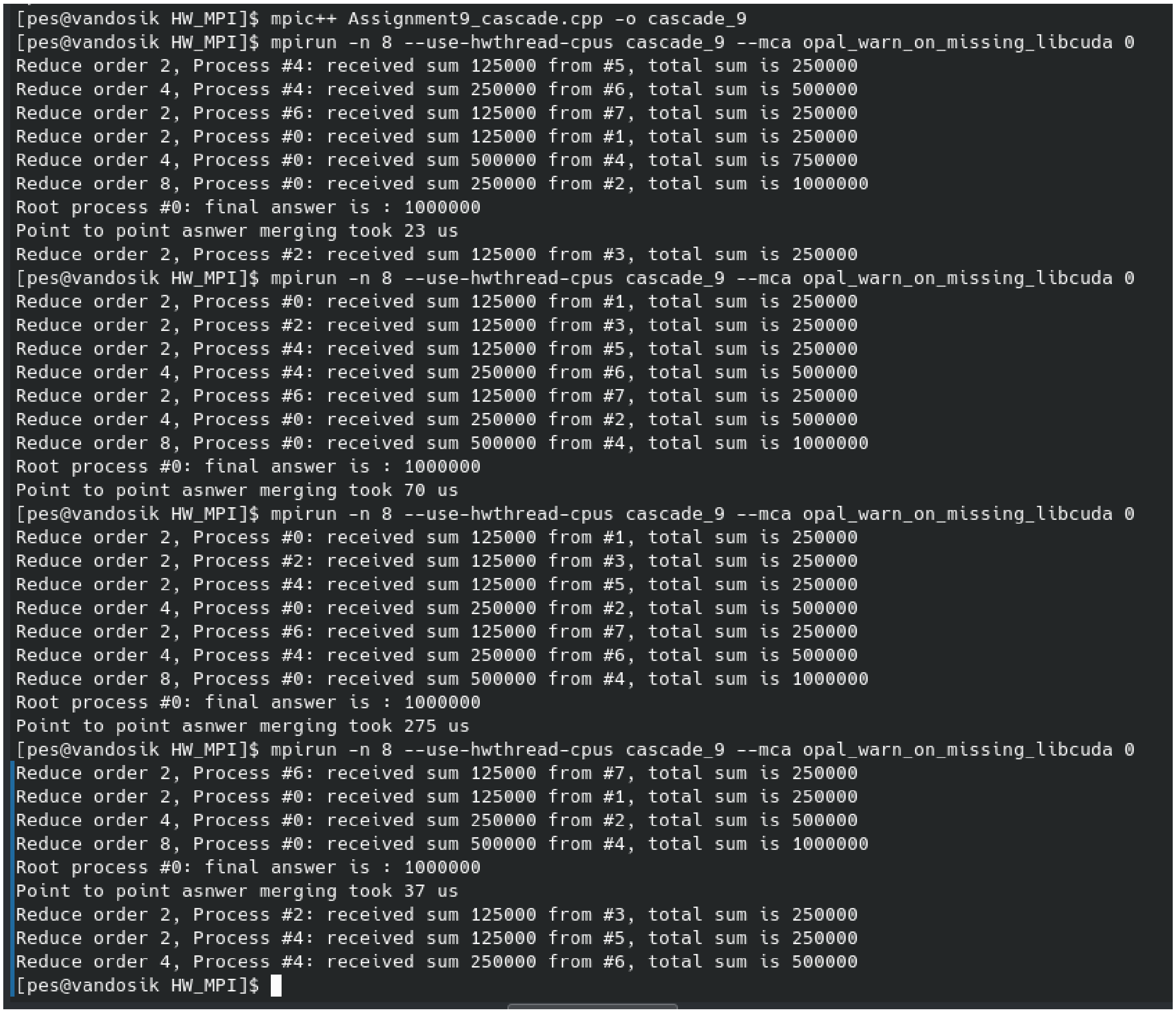
Description

The first programm implements finding array SUM using MPI_Reduce call. The second one implements cascade method, shown on the picture above. Both programms were tested four times with eight procs. The results of this testing is shown on the pictures below. As one can see, time calculations for both approaches are not stable, but all in all for Reduce variant the avarage execution is 30 percent less than for Cascade one.

Example of launch parameters and output



```
[pes@vandosik HW_MPI]$ mpic++ Assignment9_reduce.cpp -o reduce_9
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus reduce_9 --mca opal_warn_on_missing_libcuda 0
Root process #0: final answer is : 1000000
Point to point asnwer merging took 134 us
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus reduce_9 --mca opal_warn_on_missing_libcuda 0
Root process #0: final answer is : 1000000
Point to point asnwer merging took 129 us
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus reduce_9 --mca opal_warn_on_missing_libcuda 0
Root process #0: final answer is : 1000000
Point to point asnwer merging took 14 us
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus reduce_9 --mca opal_warn_on_missing_libcuda 0
Root process #0: final answer is : 1000000
Point to point asnwer merging took 16 us
```



```
[pes@vandosik HW_MPI]$ mpic++ Assignment9_cascade.cpp -o cascade_9
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus cascade_9 --mca opal_warn_on_missing_libcuda 0
Reduce order 2, Process #4: received sum 125000 from #5, total sum is 250000
Reduce order 4, Process #4: received sum 250000 from #6, total sum is 500000
Reduce order 2, Process #6: received sum 125000 from #7, total sum is 250000
Reduce order 2, Process #0: received sum 125000 from #1, total sum is 250000
Reduce order 4, Process #0: received sum 500000 from #4, total sum is 750000
Reduce order 8, Process #0: received sum 250000 from #2, total sum is 1000000
Root process #0: final answer is : 1000000
Point to point asnwer merging took 23 us
Reduce order 2, Process #2: received sum 125000 from #3, total sum is 250000
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus cascade_9 --mca opal_warn_on_missing_libcuda 0
Reduce order 2, Process #0: received sum 125000 from #1, total sum is 250000
Reduce order 2, Process #2: received sum 125000 from #3, total sum is 250000
Reduce order 2, Process #4: received sum 125000 from #5, total sum is 250000
Reduce order 4, Process #4: received sum 250000 from #6, total sum is 500000
Reduce order 2, Process #6: received sum 125000 from #7, total sum is 250000
Reduce order 4, Process #0: received sum 250000 from #2, total sum is 500000
Reduce order 8, Process #0: received sum 500000 from #4, total sum is 1000000
Root process #0: final answer is : 1000000
Point to point asnwer merging took 70 us
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus cascade_9 --mca opal_warn_on_missing_libcuda 0
Reduce order 2, Process #0: received sum 125000 from #1, total sum is 250000
Reduce order 2, Process #2: received sum 125000 from #3, total sum is 250000
Reduce order 2, Process #4: received sum 125000 from #5, total sum is 250000
Reduce order 4, Process #0: received sum 250000 from #2, total sum is 500000
Reduce order 2, Process #6: received sum 125000 from #7, total sum is 250000
Reduce order 4, Process #4: received sum 250000 from #6, total sum is 500000
Reduce order 8, Process #0: received sum 500000 from #4, total sum is 1000000
Root process #0: final answer is : 1000000
Point to point asnwer merging took 275 us
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus cascade_9 --mca opal_warn_on_missing_libcuda 0
Reduce order 2, Process #6: received sum 125000 from #7, total sum is 250000
Reduce order 2, Process #0: received sum 125000 from #1, total sum is 250000
Reduce order 4, Process #0: received sum 250000 from #2, total sum is 500000
Reduce order 8, Process #0: received sum 500000 from #4, total sum is 1000000
Root process #0: final answer is : 1000000
Point to point asnwer merging took 37 us
Reduce order 2, Process #2: received sum 125000 from #3, total sum is 250000
Reduce order 2, Process #4: received sum 125000 from #5, total sum is 250000
Reduce order 4, Process #4: received sum 250000 from #6, total sum is 500000
[pes@vandosik HW_MPI]$ 
```

Assignment 10

Complete the program Assignment10.c. Compile and run it.

Study the code carefully and explain how it works.

Listing of the program

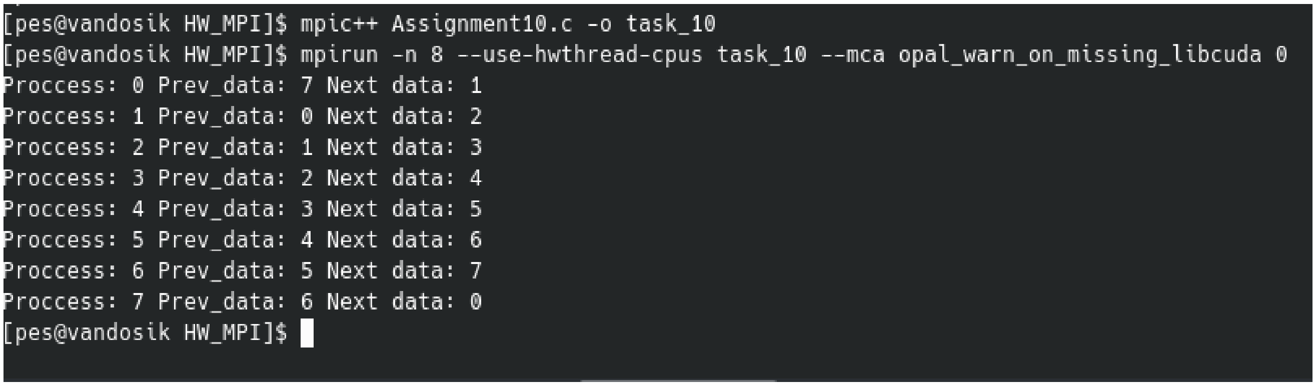
See it in my github repo

Description

Each proc sends its rank to previous and next procs. Non-blocking MPI_Irecv and MPI_Isend calls are used.

All requests completion is determined by calling the MPI_Waitall.

Example of launch parameters and output



```
[pes@vandosik HW_MPI]$ mpic++ Assignment10.c -o task_10
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus task_10 --mca opal_warn_on_missing_libcuda 0
Proccess: 0 Prev_data: 7 Next data: 1
Proccess: 1 Prev_data: 0 Next data: 2
Proccess: 2 Prev_data: 1 Next data: 3
Proccess: 3 Prev_data: 2 Next data: 4
Proccess: 4 Prev_data: 3 Next data: 5
Proccess: 5 Prev_data: 4 Next data: 6
Proccess: 6 Prev_data: 5 Next data: 7
Proccess: 7 Prev_data: 6 Next data: 0
[pes@vandosik HW_MPI]$ 
```

Assignment 8

Based onAssignment 10, write a program for ring topology exchange using the MPI_Sendrecv() function.In situations where you need to exchange data between processes, it is safer to use the overlaid MPI_Sendrecv operation. The MPI_Sendrecv function combines the execution of the send and receiveoperations. Both operations use the same communicator, but message IDs may differ. The location of thereceived and transmitted data in the address space of the process should not overlap. The data sent can beof different types and lengths.In cases when it is necessary to exchange data of the same type with replacement of the sent data with thereceived ones, it is more convenient to use the MPI_Sendrecv_replace function. In this operation, the datasent from the buf array is replaced with the received data.The special address MPI_PROC_NULL can be used for source and dest in data transfer operations.Communication operations with such an address do nothing. The use of this address is convenient instead ofusing logical constructs to analyze the conditions to send / read a message or not.

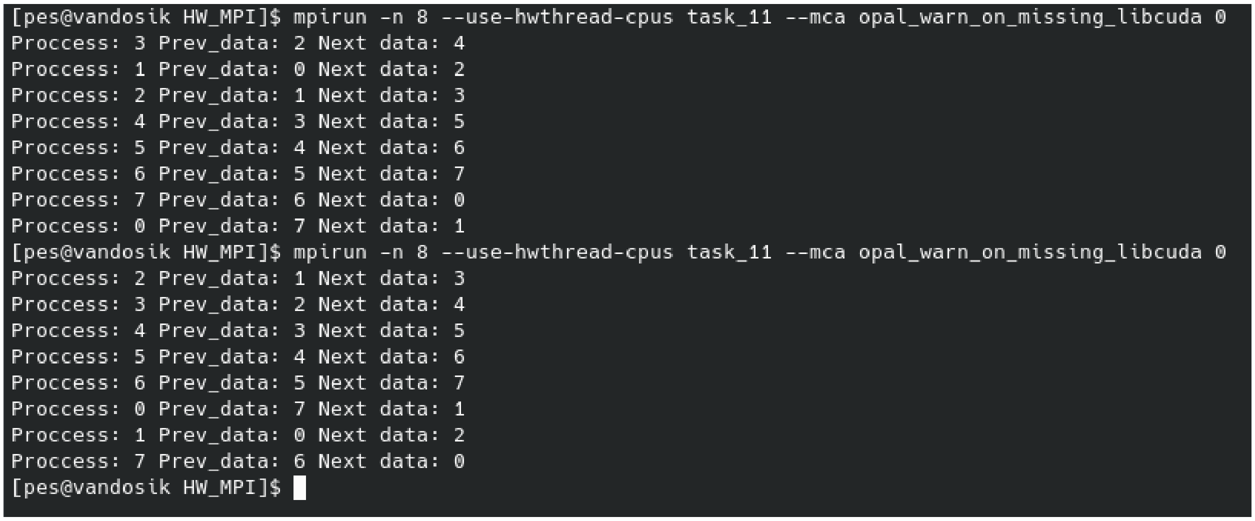
Listing of the program

See it in my github repo

Description

Program created n processes. Each process receives and sends message, equals its rank to previous and next (relatively its rank) neighbor.Main difference in new program – usage of MPI_Sendrecv function to handle sending and receiving messages. In first MPI_Sendrecv operation every processsends message to previous and receives from next. In second MPI_Sendrecv operation every processsends message to nextand receives from previous.

Example of launch parameters and output



```
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus task_11 --mca opal_warn_on_missing_libcuda 0
Proccess: 3 Prev_data: 2 Next data: 4
Proccess: 1 Prev_data: 0 Next data: 2
Proccess: 2 Prev_data: 1 Next data: 3
Proccess: 4 Prev_data: 3 Next data: 5
Proccess: 5 Prev_data: 4 Next data: 6
Proccess: 6 Prev_data: 5 Next data: 7
Proccess: 7 Prev_data: 6 Next data: 0
Proccess: 0 Prev_data: 7 Next data: 1
[pes@vandosik HW_MPI]$ mpirun -n 8 --use-hwthread-cpus task_11 --mca opal_warn_on_missing_libcuda 0
Proccess: 2 Prev_data: 1 Next data: 3
Proccess: 3 Prev_data: 2 Next data: 4
Proccess: 4 Prev_data: 3 Next data: 5
Proccess: 5 Prev_data: 4 Next data: 6
Proccess: 6 Prev_data: 5 Next data: 7
Proccess: 0 Prev_data: 7 Next data: 1
Proccess: 1 Prev_data: 0 Next data: 2
Proccess: 7 Prev_data: 6 Next data: 0
[pes@vandosik HW_MPI]$ █
```