

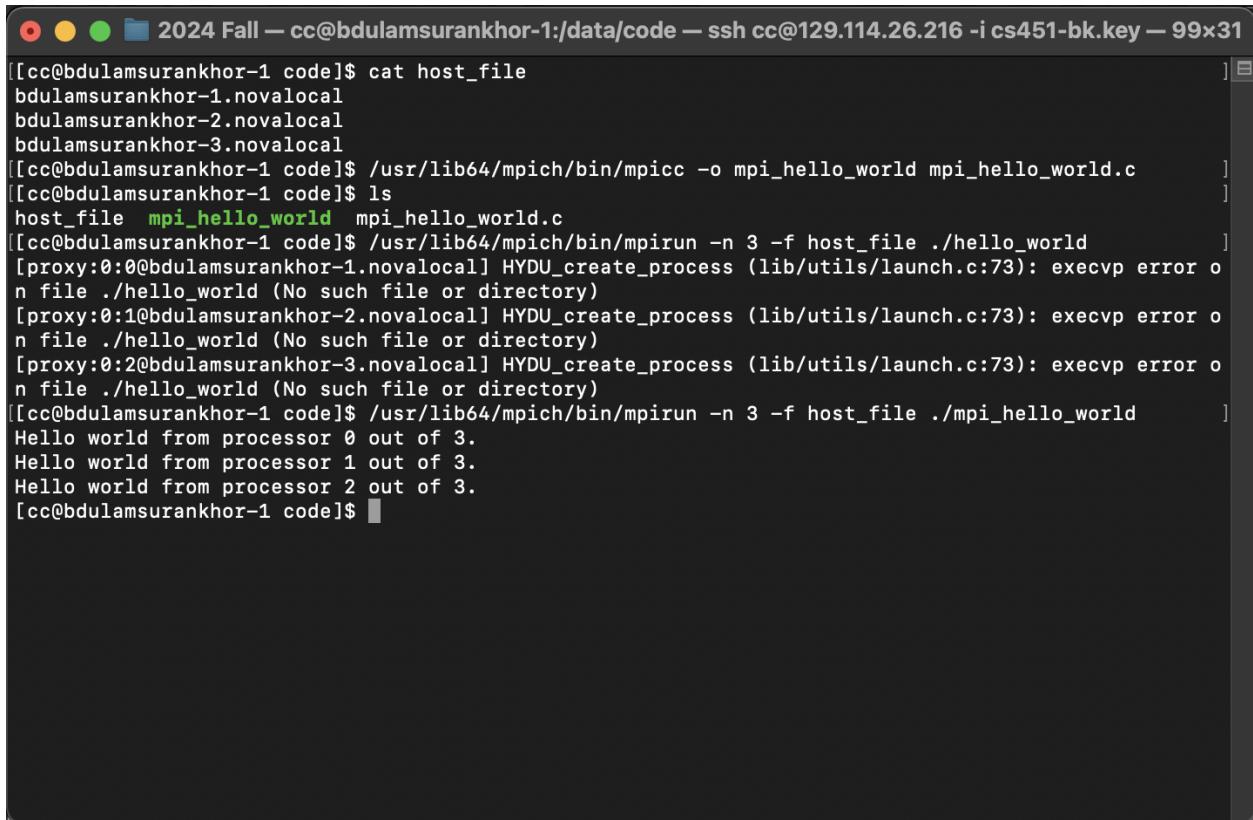
CS451 Introduction to Parallel and Distributed Computing - Assignment 4

Batkhishig Dulamsurankhor - A20543498

November 8, 2024

1. Screen shot of the apps working for part b. Screenshot of the hello world working in part a.

- (a) part a.



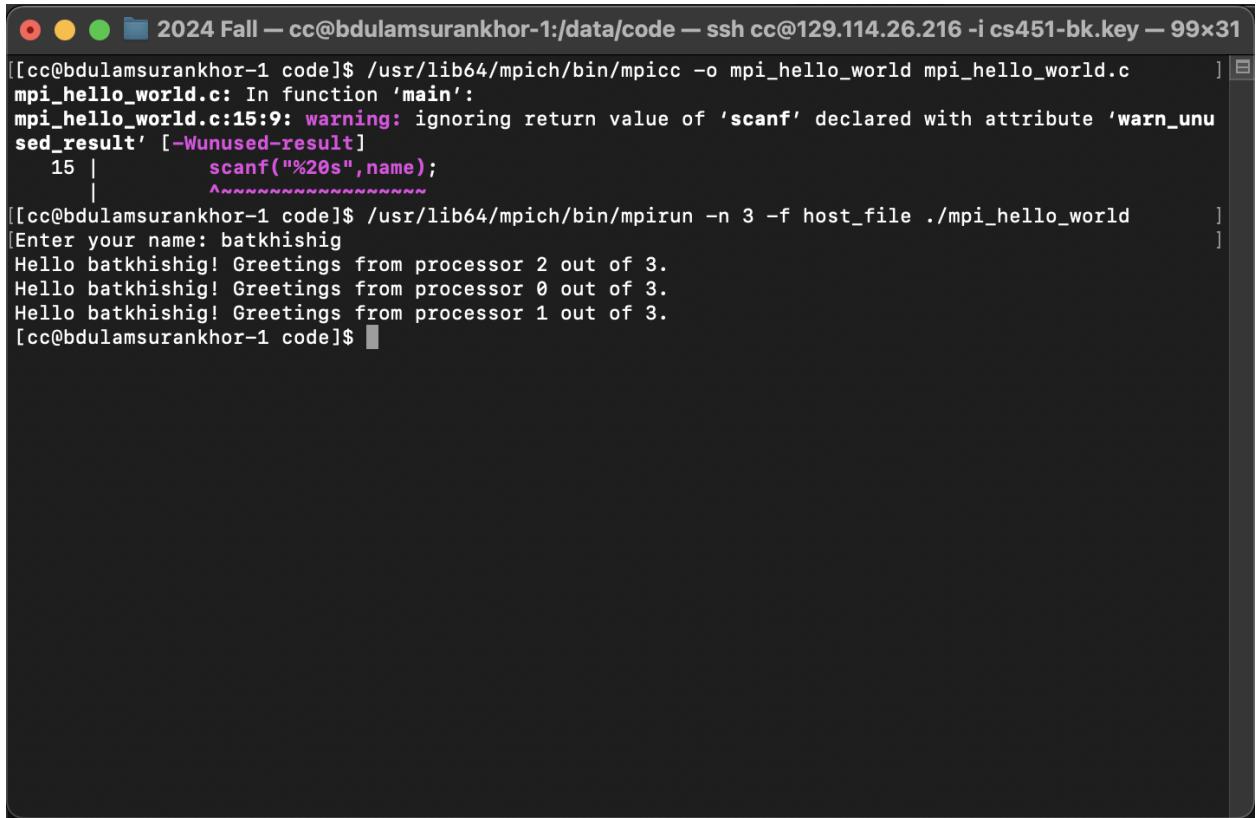
The screenshot shows a terminal window titled "2024 Fall — cc@bdulamsurankhor-1:/data/code — ssh cc@129.114.26.216 -i cs451-bk.key — 99x31". The terminal output is as follows:

```
[[cc@bdulamsurankhor-1 code]$ cat host_file
bdulamsurankhor-1.novalocal
bdulamsurankhor-2.novalocal
bdulamsurankhor-3.novalocal
[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpicc -o mpi_hello_world mpi_hello_world.c
[[cc@bdulamsurankhor-1 code]$ ls
host_file  mpi_hello_world  mpi_hello_world.c
[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 -f host_file ./hello_world
[proxy:0:0@bdulamsurankhor-1.novalocal] HYDU_create_process (lib/utils/launch.c:73): execvp error on file ./hello_world (No such file or directory)
[proxy:0:1@bdulamsurankhor-2.novalocal] HYDU_create_process (lib/utils/launch.c:73): execvp error on file ./hello_world (No such file or directory)
[proxy:0:2@bdulamsurankhor-3.novalocal] HYDU_create_process (lib/utils/launch.c:73): execvp error on file ./hello_world (No such file or directory)
[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 -f host_file ./mpi_hello_world
Hello world from processor 0 out of 3.
Hello world from processor 1 out of 3.
Hello world from processor 2 out of 3.
[[cc@bdulamsurankhor-1 code]$
```

Figure 1: Screenshot of the hello world working in part a.

- (b) part b.

- i. Adjust hello world so that after you type in your name once, when prompted by the manager node, every node salutes you, using the name you typed in.



The screenshot shows a terminal window with the following content:

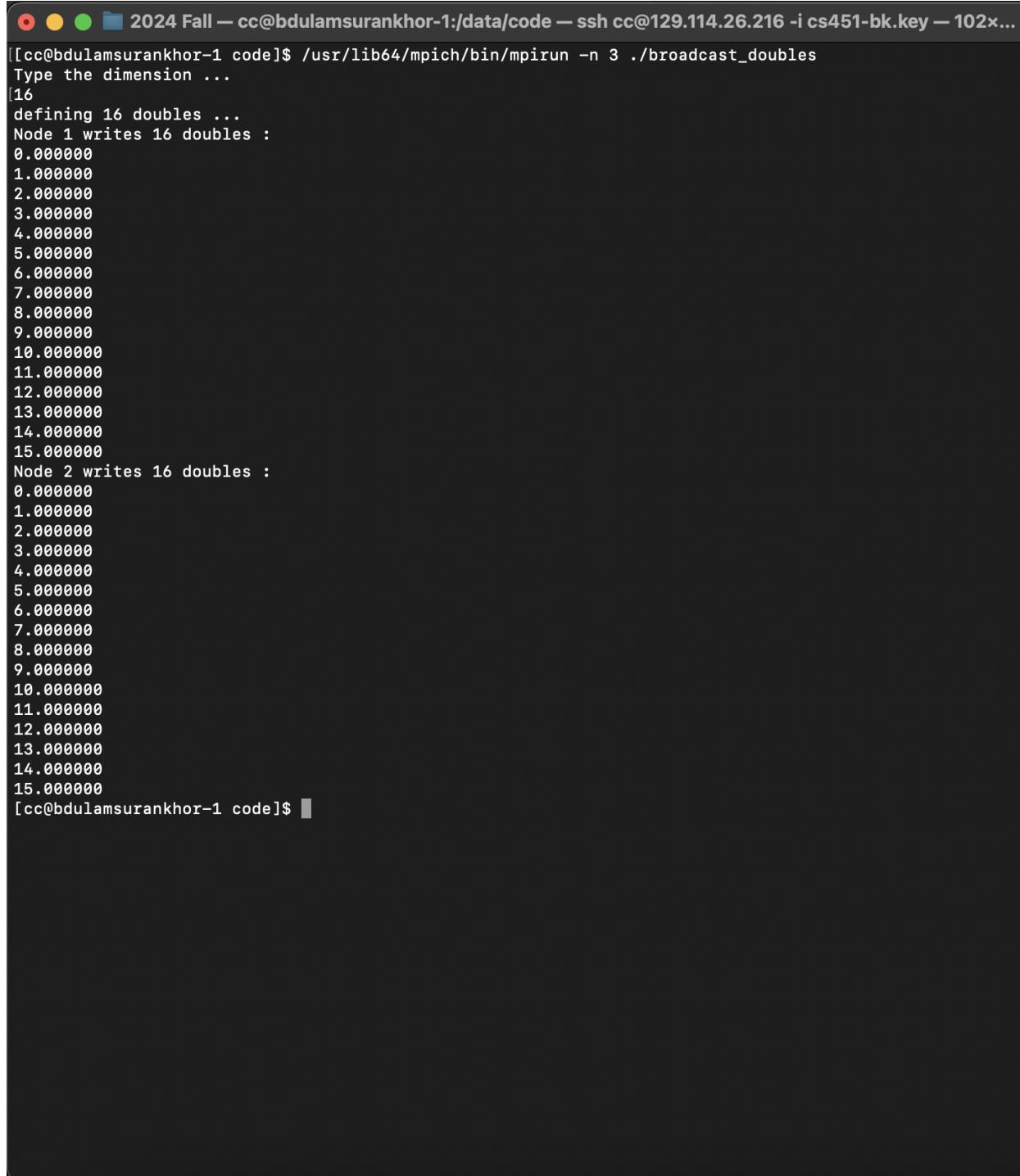
```
[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpicc -o mpi_hello_world mpi_hello_world.c
mpi_hello_world.c: In function 'main':
mpi_hello_world.c:15:9: warning: ignoring return value of 'scanf' declared with attribute 'warn_unused_result' [-Wunused-result]
  15 |     scanf("%20s", name);
      | ^~~~~~
[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 -f host_file ./mpi_hello_world
[Enter your name: batkhisig
Hello batkhisig! Greetings from processor 2 out of 3.
Hello batkhisig! Greetings from processor 0 out of 3.
Hello batkhisig! Greetings from processor 1 out of 3.
[cc@bdulamsurankhor-1 code]$
```

Figure 2: Screenshot of part b, q1.

- ii. We measure the wall clock time using time mpirun in the broadcasting of an array of doubles. To avoid typing in the dimension n, either define n as a constant in the program or redirect the input from a file that contains n. For increasing number of processes and n, investigate how the wall clock time grows. Run N as 2,4,8,16,32.

```
2024 Fall — cc@bdulamsurankhor-1:/data/code — ssh cc@129.114.26.216 -i cs451-bk.key — 102x...  
[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpicc -o broadcast_doubles broadcast_doubles.c  
broadcast_doubles.c: In function 'main':  
broadcast_doubles.c:24:7: warning: ignoring return value of 'scanf' declared with attribute 'warn_unused_result' [-Wunused-result]  
 24 |     scanf("%d",&n);  
   | ^~~~~~  
[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 ./broadcast_doubles  
Type the dimension ...  
2  
defining 2 doubles ...  
Node 1 writes 2 doubles :  
0.000000  
1.000000  
Node 2 writes 2 doubles :  
0.000000  
1.000000  
[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 ./broadcast_doubles  
Type the dimension ...  
4  
defining 4 doubles ...  
Node 1 writes 4 doubles :  
0.000000  
1.000000  
2.000000  
3.000000  
Node 2 writes 4 doubles :  
0.000000  
1.000000  
2.000000  
3.000000  
[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 ./broadcast_doubles  
Type the dimension ...  
8  
defining 8 doubles ...  
Node 1 writes 8 doubles :  
0.000000  
1.000000  
2.000000  
3.000000  
4.000000  
5.000000  
6.000000  
7.000000  
Node 2 writes 8 doubles :  
0.000000  
1.000000  
2.000000  
3.000000  
4.000000  
5.000000  
6.000000  
7.000000  
[cc@bdulamsurankhor-1 code]$
```

Figure 3: Screenshot of part b, q2. n=2,4,8



```
2024 Fall — cc@bdulamsurankhor-1:/data/code — ssh cc@129.114.26.216 -i cs451-bk.key — 102x...  
[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 ./broadcast_doubles  
Type the dimension ...  
[16  
defining 16 doubles ...  
Node 1 writes 16 doubles :  
0.000000  
1.000000  
2.000000  
3.000000  
4.000000  
5.000000  
6.000000  
7.000000  
8.000000  
9.000000  
10.000000  
11.000000  
12.000000  
13.000000  
14.000000  
15.000000  
Node 2 writes 16 doubles :  
0.000000  
1.000000  
2.000000  
3.000000  
4.000000  
5.000000  
6.000000  
7.000000  
8.000000  
9.000000  
10.000000  
11.000000  
12.000000  
13.000000  
14.000000  
15.000000  
[cc@bdulamsurankhor-1 code]$
```

Figure 4: Screenshot of part b, q2. n=16

```
2024 Fall — cc@bdulamsurankhor-1:/data/code — ssh cc@129.114.26.216 -i cs451-bk.key — 102x...  
[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 ./broadcast_doubles  
Type the dimension ...  
|32  
defining 32 doubles ...  
Node 2 writes 32 doubles :  
0.000000  
1.000000  
2.000000  
3.000000  
4.000000  
5.000000  
6.000000  
7.000000  
8.000000  
9.000000  
10.000000  
11.000000  
12.000000  
13.000000  
14.000000  
15.000000  
16.000000  
17.000000  
18.000000  
19.000000  
20.000000  
21.000000  
22.000000  
23.000000  
24.000000  
25.000000  
26.000000  
27.000000  
28.000000  
29.000000  
30.000000  
31.000000  
Node 1 writes 32 doubles :  
0.000000  
1.000000  
2.000000  
3.000000  
4.000000  
5.000000  
6.000000  
7.000000  
8.000000  
9.000000  
10.000000  
11.000000  
12.000000  
13.000000  
14.000000  
15.000000  
16.000000  
17.000000  
18.000000
```

Figure 5: Screenshot of part b, q2. n=32

```
2024 Fall — cc@bdulamsurankhor-1:/data/code — ssh cc@129.114.26.216 -i cs451-bk.key — 102x...  
9.000000  
10.000000  
11.000000  
12.000000  
13.000000  
14.000000  
15.000000  
16.000000  
17.000000  
18.000000  
19.000000  
20.000000  
21.000000  
22.000000  
23.000000  
24.000000  
25.000000  
26.000000  
27.000000  
28.000000  
29.000000  
30.000000  
31.000000  
Node 1 writes 32 doubles :  
0.000000  
1.000000  
2.000000  
3.000000  
4.000000  
5.000000  
6.000000  
7.000000  
8.000000  
9.000000  
10.000000  
11.000000  
12.000000  
13.000000  
14.000000  
15.000000  
16.000000  
17.000000  
18.000000  
19.000000  
20.000000  
21.000000  
22.000000  
23.000000  
24.000000  
25.000000  
26.000000  
27.000000  
28.000000  
29.000000  
30.000000  
31.000000  
[cc@bdulamsurankhor-1 code]$
```

Figure 6: Screenshot of part b, q2. n=32

- iii. Compile and run both static_loaddist.c and dynamic_loaddist.c and compare the runtimes.
Run N as 2,4,8,16,32.

```

2024 Fall — cc@bdulamsurankhor-1:/data/code — ssh cc@129.114.26.216 -i...
[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpicc -o static_loaddist static_loaddist.c
static_loaddist.c: In function 'main':
static_loaddist.c:32:19: warning: ignoring return value of 'scanf' declared with attribute 'warn_unused_result' [-Wunused-result]
    32 |     int nbjobs; scanf("%d",&nbjobs);
|     ~~~~~
[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 -f host_file ./static_loaddist
reading the #jobs per compute node...
2
sending 0 to 1
sending 1 to 2
node 1 received 0
-> 1 computes b
node 1 sends b
received b from 1
received c from 2
sending 2 to 1
sending 3 to 2
node 2 received 1
-> 2 computes c
node 2 sends c
node 2 received 3
-> 2 computes c
node 2 sends c
node 1 received 2
-> 1 computes b
node 1 sends b
received b from 1
received c from 2
sending -1 to 1
sending -1 to 2
The result : bcbc
node 2 received -1
node 1 received -1
[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 -f host_file ./static_loaddist
reading the #jobs per compute node...
4
sending 0 to 1
sending 1 to 2
node 1 received 0
-> 1 computes b
node 1 sends b
received message from 1
received b from 1
sending 2 to 1
received message from 2
received c from 2
#jobs done : 1
sending 2 to 1
received message from 2
received c from 2
#jobs done : 2
sending 3 to 2
received message from 1
received b from 1
#jobs done : 3
sending -1 to 1
node 1 received 2
-> 1 computes b
node 1 sends b
node 2 received 2
-> 2 computes c
node 2 sends c
node 2 received 3
-> 2 computes c
node 2 sends c
received message from 2
node 1 received -1
received c from 2
#jobs done : 4
sending -1 to 2
The result : bcbc
node 2 received -1

```

Figure 7: Screenshot of part b, q3. n=2

```

2024 Fall — cc@bdulamsurankhor-1:/data/code — ssh cc@129.114.26.216 -i...
received c from 2
sending -1 to 1
sending -1 to 2
The result : bcbc
node 2 received -1
node 1 received -1
[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 -f host_file ./stat1
c loaddist
reading the #jobs per compute node...
4
sending 0 to 1
sending 1 to 2
node 1 received 0
-> 1 computes b
node 1 sends b
received b from 1
received c from 2
sending 2 to 1
sending 3 to 2
node 2 received 1
-> 2 computes c
node 2 sends c
received b from 1
node 1 received 2
-> 1 computes b
node 1 sends b
node 2 received 3
-> 2 computes c
node 2 sends c
received c from 2
sending 4 to 1
sending 5 to 2
received b from 1
node 2 received 5
-> 2 computes c
node 2 sends c
received c from 2
node 1 received 4
-> 1 computes b
node 1 sends b
sending 6 to 1
sending 7 to 2
received b from 1
node 1 received 6
-> 1 computes b
node 1 sends b
received c from 2
node 2 received 7
-> 2 computes c
node 2 sends c
sending -1 to 1
sending 4 to 2
The result : bcbcbcbc
node 1 received -1
node 2 received -1
[cc@bdulamsurankhor-1 code]$ 

batkhishig — cc@bdulamsurankhor-1:/data/code — ssh cc@129.114.26.216 -i c...
received b from 1
#jobs done : 1
sending 2 to 1
node 1 received 2
-> 1 computes b
node 1 sends b
node 2 received 2
received message from 1
-> 2 computes c
received b from 1
node 2 sends c
#jobs done : 2
node 1 received 3
-> 1 computes b
node 1 sends b
node 2 receives 3
-> 2 computes c
received message from 2
node 2 sends c
received c from 2
node 1 received 7
-> 1 computes b
node 1 sends b
node 2 received 6
#jobs done : 3
-> 2 computes c
sending 4 to 1
node 1 received -1
node 2 sends c
received message from 1
node 2 received -1
received b from 1
#jobs done : 4
sending 5 to 1
received message from 2
received c from 2
#jobs done : 5
sending 6 to 2
received message from 1
received b from 1
#jobs done : 6
sending 7 to 1
received message from 2
received c from 2
#jobs done : 7
sending -1 to 2
received message from 1
received b from 1
#jobs done : 8
sending 4 to 1
The result : bcbcbcbc
[cc@bdulamsurankhor-1 code]$ 

```

Figure 8: Screenshot of part b, q3. n=4

```

2024 Fall — cc@bdulamsurankhor-1:/data/code — ssh cc@129.114.26.216 -i ...
node 2 sends c
sending 7 to 2
received b from 1
received c from 2
sending 8 to 1
node 1 received 6
-> 1 computes b
node 1 sends b
node 2 received 7
-> 2 computes c
node 2 sends c
sending 9 to 2
received b from 1
received c from 2
sending 10 to 1
node 1 received 8
-> 1 computes b
node 1 sends b
node 2 received 9
-> 2 computes c
node 2 sends c
sending 11 to 2
node 2 received 11
-> 2 computes c
node 2 sends c
node 1 received 10
-> 1 computes b
node 1 sends b
received b from 1
received c from 2
sending 12 to 1
node 2 received 12
-> 2 computes c
node 2 sends c
node 1 received 13
-> 1 computes b
node 1 sends b
received b from 1
received c from 2
sending 13 to 1
node 2 received 13
-> 2 computes c
node 2 sends c
received b from 1
received c from 2
sending 14 to 1
node 1 received 12
-> 1 computes b
node 1 sends b
sending 15 to 2
node 2 received 15
-> 2 computes c
node 2 sends c
received b from 1
received c from 2
sending -1 to 1
node 1 received 14
The result : bccbccbccbccbccbc
node 1 received 14
-> 1 computes b
node 1 sends b
node 2 received -1
node 1 received -1
[cc@bdulamsurankhor-1 code]$ 

batkhishig — cc@bdulamsurankhor-1:/data/code — ssh cc@129.114.26.216 -i c...
received c from 2
#jobs done : 9
sending 10 to 2
node 1 received 9
-> 1 computes b
node 1 sends b
received message from 1
node 2 received 10
-> 2 computes c
node 2 sends c
received b from 1
#jobs done : 10
sending 11 to 1
received message from 2
received c from 2
#jobs done : 11
sending 12 to 2
received message from 1
received b from 1
#jobs done : 12
sending 13 to 1
node 1 received 11
-> 1 computes b
node 1 sends b
node 2 received 12
-> 2 computes c
node 2 sends c
received message from 2
received c from 2
#jobs done : 13
sending 14 to 2
received message from 1
received b from 1
#jobs done : 14
sending 15 to 1
node 1 received 13
-> 1 computes b
node 1 sends b
node 2 received 14
-> 2 computes c
node 2 sends c
received message from 2
received c from 2
#jobs done : 15
sending -1 to 2
node 1 received 15
-> 1 computes b
node 1 sends b
node 2 received -1
received message from 1
received b from 1
#jobs done : 16
sending -1 to 1
The result : hcchccbcbcbcbcb
node 1 received -1
[cc@bdulamsurankhor-1 code]$ 

```

Figure 9: Screenshot of part b, q3. n=8

Figure 10: Screenshot of part b, q3. n=16

Figure 11: Screenshot of part b, q3. n=32

- iv. Adjust parallel_sum.c to go to 200 instead of 100 and adjust it to work for p processors where the dimension n of the array is a multiple of p. In other words, so if you have N = 3 or 6 , it'll work correctly.

```

2024 Fall — cc@bdulamsurankhor-1:/data/code — ssh cc@129.114.26.216 -i cs451-bk.key — ...
[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpicc -o parallel_sum parallel_sum.c
parallel_sum.c: In function 'main':
parallel_sum.c:29:10: warning: this 'for' clause does not guard... [-Wmisleading-indentation]
  29 |         for (j=0; j<200; j++) printf("%d",data[j]); printf("\n");
     |         ^
parallel_sum.c:29:55: note: ...this statement, but the latter is misleadingly indented as if it were guarded by the 'for'
  29 |         for (j=0; j<200; j++) printf("%d",data[j]); printf("\n");
     |         ^
[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 3 ./parallel_sum
The data to sum :  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 6
2 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94
95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 1
20 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 14
4 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168
169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192
193 194 195 196 197 198 199 200
Node 0 has numbers to sum : 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 5
9 60 61 62 63 64 65 66 67
Node 0 computes the sum 2278
Node 2 has numbers to sum : 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 1
52 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 17
6 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200
Node 2 computes the sum 11055
Node 1 has numbers to sum : 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90
91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 11
7 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134
Node 1 computes the sum 6767
The total sum : 20100, which should be 20100.
[cc@bdulamsurankhor-1 code]$ 

```

Figure 12: Screenshot of part b, q4. n=3

```

[[cc@bdulamsurankhor-1 code]$ /usr/lib64/mpich/bin/mpirun -n 6 ./parallel_sum
The data to sum : 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 6
2 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94
95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 1
20 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 14
4 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168
169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192
193 194 195 196 197 198 199 200
Node 0 has numbers to sum : 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
27 28 29 30 31 32 33 34
Node 0 computes the sum 595
Node 3 has numbers to sum : 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 1
19 120 121Node 1 has numbers to sum : 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 12
2 123 124 125 126 127 128 129 130 131 132 133 134
Node 3 computes the sum 3894
54 55 56 57 58 59 60 61 62 63 64 65 66 67 68
Node 1 computes the sum 1751
Node 2 has numbers to sum : 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91
92 93 94 95 96 97 98 99 100 101
Node 2 computes the sum 2805
Node 5 has numbers to sum : 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 1
85 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200
Node 5 computes the sum 6072
Node 4 has numbers to sum : 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 1
52 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167
Node 4 computes the sum 4983
The total sum : 20100, which should be 20100.
[cc@bdulamsurankhor-1 code]$ 
```

Figure 13: Screenshot of part b, q4. n=6

2. Write up answer the following - a few sentences each:

- (a) What was the purpose of the NFS server? What happens if you run hello world from a non-shared spot?

The purpose of the NFS server is to provide a shared file system across the nodes. This ensures

consistency of data shared between mpi nodes. If we run mpi hello world from a non-shared spot, nodes can't access the same file directly, and might attempt to run an independent instance of the program from their local storage.

- (b) How are the different instances of the same program communicating?

They communicate through message-passing using functions: MPI_Send, MPI_Recv and MPI_Reduce etc. MPI_Send and MPI_Recv is used for point-to-point communication. These functions manage data transfer between nodes and ensure that all processes are synchronized for their tasks.

- (c) How might the interconnect effect the speed of a MPI program?

The network connection between nodes significantly affects the speed of MPI program. Because MPI relies on data communication between processes running on different nodes. Bandwidth, Latency, Congestion and Overheads have direct impact on the performance of the MPI program.