FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION OF HIGHER EDUCATION

ITMO UNIVERSITY

Report

MPI. Assignments $20 - 21$

Parallel algorithms for the analysis and synthesis of data

Performed by
Aleksandr Shirokov
J4133c
Accepted by
Petr Andriushchenko
Deadline: 26.12.21

St. Petersburg
2021

Contents

1 Assignments

1.1 Assignment 20. MPI. Parallel I O. Working with files. Access to data. Buffered reading from a file

1.1.1 Formulation of the problem

Understand the new functions in Assignment20.c, complete the program according to the assignment, explain the execution of the program.

Write a function that will create a file **"file.txt"** with random content (or with specific text). The function must be executed before the program reads the contents of the file. Run the program on one process. Check if the contents of the file are displayed correctly. Add an option that will delete the file on close

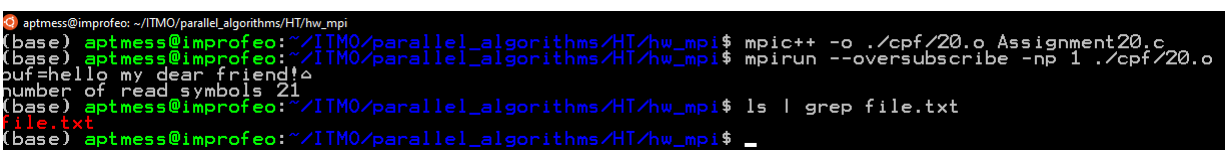
1.1.2 Example of launch parameters and output. Detailed description of solution

Code for **assignment 20** is here.

Compilation example: mpic++ -o ./cpf/20.o Assignment20.c

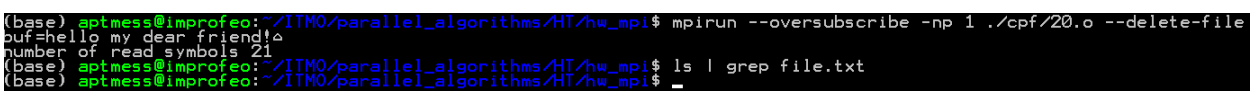
Launch example:

- mpirun –oversubscribe -np 1 ./cpf/20.o - without delete option



File exists and also their is input

- mpirun –oversubscribe -np 1 ./cpf/20.o –delete-file - with delete option



As we can see the file is deleted.

Let's move to the the code and explain how it works.

```
1   #include <stdio.h>
2   #include <iostream>
3   #include "mpi.h"
4   #define BUFSIZE 100
5
6   using namespace std;
7
8
9   int main(int argc, char **argv)
10  {
11      int bufsize, num, sum;
12
13      MPI_Init(&argc, &argv);
14
15      MPI_Status status;
16      MPI_File fh;
17
18      char buf[BUFSIZE];
19      const char* filename = "file.txt";
20      string message ("hello my dear friend!");
21
22      int len = message.length();
23      const char *cstr = message.c_str();
24
25      // start of writing to file "file.txt" with text "Hello world!"
26      MPI_File_open(MPI_COMM_WORLD, filename, MPI_MODE_CREATE | MPI_MODE_WRONLY, MPI_INFO_NULL, &fh);
27      MPI_File_write(fh, cstr, len, MPI_CHAR, &status);
28      MPI_File_close(&fh);
29      // end of writing to file
30
31      MPI_File_open(MPI_COMM_WORLD, filename, MPI_MODE_RDONLY, MPI_INFO_NULL, &fh);
32      MPI_File_set_view(fh, 0, MPI_CHAR, MPI_CHAR, "native", MPI_INFO_NULL);
33      sum = 0;
34      do {
35          MPI_File_read(fh, buf, BUFSIZE, MPI_CHAR, &status);
36          MPI_Get_count(&status, MPI_CHAR, &num);
37          printf("buf=%s\n", buf);
38          sum += num;
39      } while (num >= BUFSIZE);
40      MPI_File_close(&fh);
41      //print the number of read symbols sum from the file
42      printf("number of read symbols %d\n", sum);
43      if ((argc > 1) && (string(argv[1]) == "--delete-file"))
44      {
45          MPI_File_delete(filename, MPI_INFO_NULL);
46      }
47      MPI_Finalize();
48  }
```

Assignment20 code

In this lab there are new functions of MPI that works with some IO (input and output) operations with files. In lines $19 - 20$ there is a filename and string which will be filled in file, in line $25 - 29$ there is a function that open files (MPI_File_open), write in file message (MPI_File_write) and standard closing MPI_File_Close. After that weopen file and read only mode, MPI_File_set_view function changes process's view of data in file (collective) and in lines $34 - 39$ while end on the file count the amount of symbols in file. File closing in line 40, printing the number of read sybmols sum from the file and if there is a option –delete-file there is function to delete file. Program works correctly for each option of running program as we checking it with grep higher.

1.2 Assignment 21. MPI. Parallel I O. Working with files. Access to data. Collective reading from a file

1.2.1 Formulation of the problem

Understand the new functions in Assignment21.c, complete the program according to the assignment, explain the execution of the program.

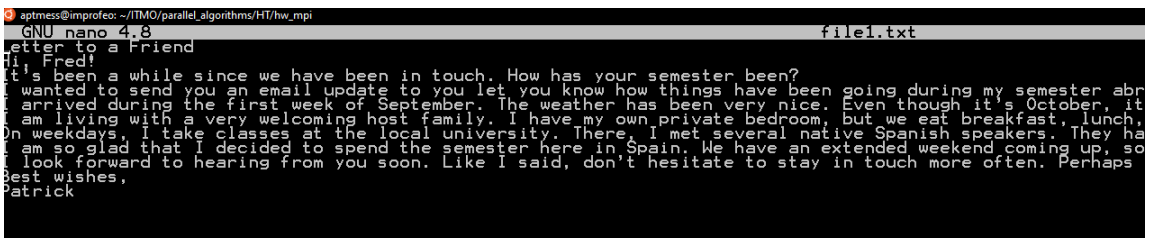
Create a file and fill it with bulky text, output the content in parallel. Change the step of reading the contents of the file and the number of characters to be output by each process.

1.2.2 Example of launch parameters and output. Detailed description of solution

Code for **assignment 21** is here.

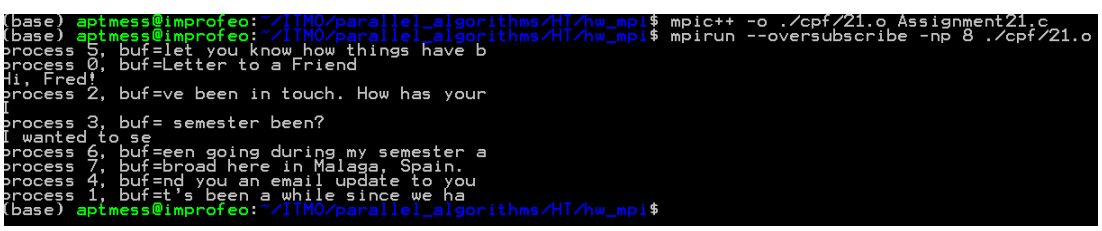
Compilation example: mpic++ -o ./cpf/21.o Assignment21.c

Launch example:



It is a text in file **file1.txt**

Run program with 10 sybmols per process. Let's change amount of sybmols to 30



Amount of sybmols per process is 30.

Let's move to the the code and explain how it works.

```c
#include <stdio.h>
#include "mpi.h"
int main(int argc, char **argv)
{
    int rank;
    MPI_Init(&argc, &argv);
    MPI_File fh;
    char buf[30];
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_File_open(MPI_COMM_WORLD, "file1.txt", MPI_MODE_RDONLY, MPI_INFO_NULL, &fh);
    MPI_File_set_view(fh, rank * 30, MPI_CHAR, MPI_CHAR, "native", MPI_INFO_NULL);
    MPI_File_read_all(fh, buf, 30, MPI_CHAR, MPI_STATUS_IGNORE);
    printf("process %d, buf=%s\n", rank, buf);
    MPI_File_close(&fh);
    MPI_Finalize();
}
```

Assignment21 code

As we can see our program read file **file1.txt** in parallel mode by fixed amount of sybmols (i have increased amount of sybmols from 10 to 30 as it is shown higher) for each process there is a formula which sybmols each process is reading using function MPI_File_set_view - $[(20 \cdot \text{rank}) \ldots (20 \cdot (\text{rank} + 1) - 1)]$. As we can see program works correctly and we can control with parallel mode the velocity of reading files - it is very common operation in MPI i think. That's all!

1.3 Appendix

The link to the sourse code which is placed on my github.