CSE 344 HOMEWORK 2 REPORT

EMRE SEZER
1901042640

First, I read the input file with readLine() function. This function reads 3 for each coordination. For each R_i it reads total of 30 bytes. Input file stores continuous characters. Which means that, there is no special characters between each 30 bytes between R_i's. If your example input file includes special characters between them (like: ',', ' ' or '\n') you need to delete them and make the file 1 continuous line. I attached an example input file at the folder.

After each 30 bytes are read it creates a child process. Each child process calls execve() function with environmental variables. execve function's parameters are passed like required at the homework's pdf file. execve() function is connected with the R_i.c file.

At the main function in the R_i.c file:

It calculates the covariant matrix and writes to the output file. Output file will include i times of lines (for each R_i). Each line includes 10 numbers separated with ',' characters. First 9 numbers are covariant matrix and 10'th number is for R_i's custom id (for showing which which R process is it). I attached an example output file at the folder.

After loop is terminated which is for creating child processes. Parent process waits for child processes to be terminated.Then, it reads output file and calculates Frobenius Norm for each R_i. In the end it finds the closest 2 matrixes and prints them with the distance.

Signal Handling:

I control if SIGINT signal has come at the while loop before creating a child process. If that is happened, I kill all of the child processes, free the allocated memory and close the files.

At the R_i.c:

I control if that is happened right before writing to the output file. At that situation, I free the allocated memory and close the file.

Notes:

At the R_i.c I lock output file with fcntl() function and F_LCKW. I do that. Because, I don't want 2 or more child processes writing at the output file at the same time. With that approach I handle that situation.

If the other child process tries to reach to the output file. It waits for it to unlock it.

I create a clear output file before creating child processes.

Running the homework:

You need to type ./process before the arguments

For example you can type:

./processP -i input.txt -o output.txt