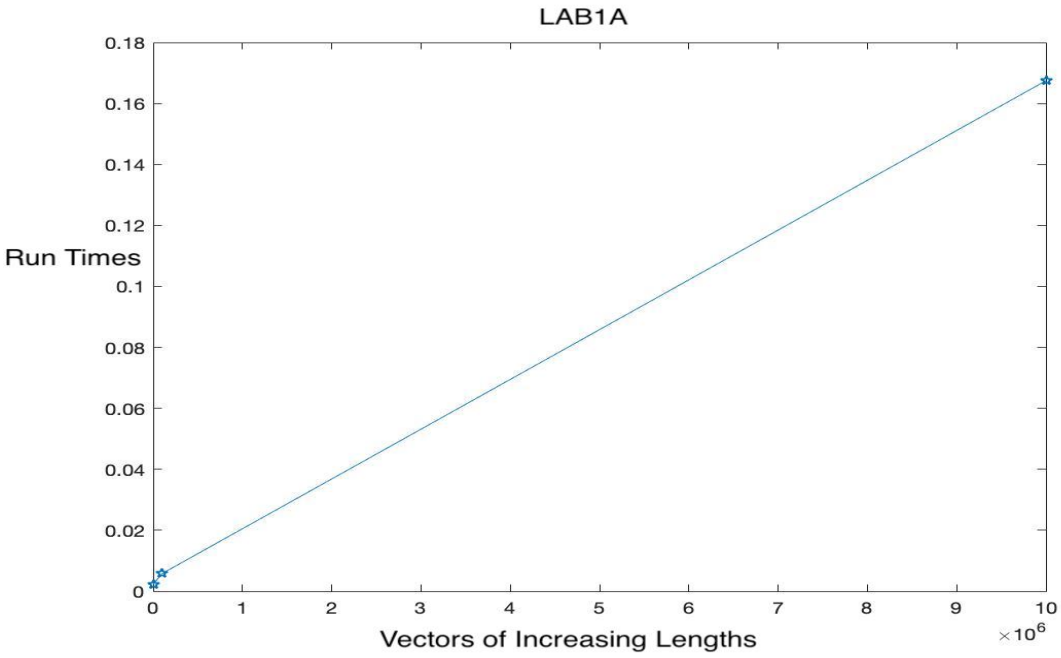
John Do
CSC 452-001
Dr. Droz
September 25, 2019

Lab Assignment: Lab 1

1.



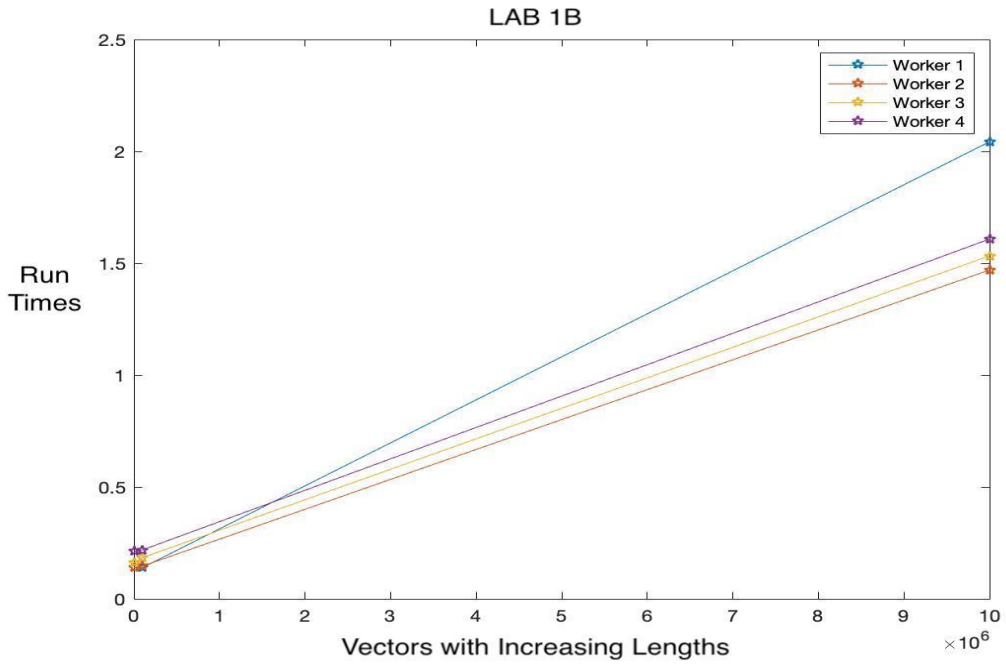
a.

Vectors of Increasing Length	Run Times
1000	0.0023116
100000	0.0058158
10000000	0.1674582

b. The number of dimensions used in this lab was 2.

John Do
CSC 452-001
Dr. Droz
September 25, 2019

2.



a.

		10^3		
Workers	1	2	3	4
Run Times	0.13956	0.13972	0.16154	0.21186

b.

		10^5		
Workers	1	2	3	4
Run Times	0.13976	0.1464	0.184	0.2187

c.

		10^7		
Workers	1	2	3	4
Run Times	2.0428	1.4694	1.5328	1.6079

d. Since I can only use 4 workers, my data shows that two workers would be the greatest speedup because the idea is that there reaches a certain point where adding more workers will not equate to increase the speed; therefore, the overhead for managing the workers will take up more time than it saves. In the data, from 10^3, 10^5, and 10^7, 2 workers are quicker but it does not require too much overhead and you could also use 3 workers as well; however, it does

not make much of a difference since they are fairly close to each other in time. It just depends on how much management from the overhead to the workers.

e. The number of dimensions is 2 for this lab experiment.

3. To parallelize this loni script, you change the for loop to how many workers are being used to run the lab such as 1-20 in this case. It would distribute the job to each of the incrementing numbers of workers to demonstrate the time it will get the job done. On the loni cluster, you will use a submit script to call the LAB1C program that has the iteration of 1:20 threads with the command "**qsub start -A cyen_loni405**". The lab1C will iterate 1:20 threads to call the LAB1B. This will return the time of different numbers of threads. You can only use this script if the lab function you are calling can generate a number of different workers. Since the fastest time that was generated was "4.2442" with 7 workers. Since there is 1 thread per core in the loni cluster with 2 sockets as well, the ideal computational core is 7 in this case. The optimal threads would be 14.

4. The input data elements are partitioned by using the smpd function where lets you define a block of code to run simultaneously on multiple workers. If the user runs one program in the Matlab client, and those parts of it labeled as spmd blocks run on the workers. When the spmd block is complete, the program continues running in the client. The job manager on the local machine will distribute the input data between different workers in the pool. Each worker has their own indexes, which can be called by the client when the workers are done with their processes. Each lab determines its start and end indices by replicating the same spmd code. Once the spmd command starts, it inputs the data in every worker with the same code to generate an output back to the job manager. The "multiple data" aspect means that even though the spmd statement runs identical code on all workers, each worker can have different, unique data for that code. It shows speed up because spmd lets several workers compute solutions simultaneously or for large data sets spmd lets the data be distributed to multiple workers. Typical applications appropriate for spmd are those that require running simultaneous execution of a program on multiple data sets. This is when communication and synchronization are required between the workers.

5. ***Turned in the Matlab code for this portion***

6.
    a. Yes
    b. Yes
    c. Yes
    d. Yes

John Do
CSC 452-001
Dr. Droz
September 25, 2019

Vectors of Increasing Lengths	Run Times
10^3	0.17324
10^5	0.07876
10^7	1.5332

e. This method of implementing ring communications would produce speedup because if we look at the other labs with their time. Spmd lets several workers compute solutions simultaneously or for large data sets spmd lets the data be distributed to multiple workers. The times that we produce using the ring communication is slightly faster when comparing to the second lab portion, Lab1B. For example, in the ring communication, you are running each function and sending the result to another lab index, whereas the labs are just running one function with a specific number of workers and increasing lengths of a vector. The disadvantage of this would be that if one function is not working in the ring communication, it would ruin the whole program. The data must be precise and relevant in order to send this data to the next lab index. A situation that will be beneficial for using ring communications is using complex algorithms because it would make it easier to understand where the problem lies like if a calculation is wrong, it would be easy to locate it. Another benefit of using a ring communication is on the network or in a cluster. It would help pass information around to one place to another in order to get the information.

John Do
CSC 452-001
Dr. Droz
September 25, 2019

Lab Report: Lab 1

I. Introduction:

The purpose of this lab is to get the basic fundamentals of Matlab. As a class, we are using basic matlab functions to create algorithms, parallelisms, and ring communications between workers. We will be doing experiments on the local machine and on the loni cluster to generate data to help us answer questions and plot run times from vectors with increasing length. This will help us get an understanding how data is being generated by algorithms and transmitted between worker nodes. The structure of this report is to implement different methods and algorithms on matlab within our local machine or loni cluster. Some of the problems that may need to be considered is the time that is being conducted by each worker node. Is it going to take a long time depending on the computational core or will the worker nodes can evenly distribute different data between each other. After implementing different methods and algorithms, experiments will be conducted and reported on different values being inputted and get an understanding on how the outcome is generated. Results will be stated after conducting experiments and demonstrating how the outcome was gotten. At the end of the report will be the conclusions and future work, summarizing the whole report and determining how this lab will help us in the future when conducting similar experiments.

II. Methods & Algorithms:

The metrics of importance to our distributed or cloud systems are time, availability, computational cores. During this lab, we are timing the generated output from different algorithms and functions to determine different factors of the lab. For example, we are timing number of workers' output to see if it is faster to generate output or cause too much overhead to manage. Another thing that we are using is the loni cluster, which should be available to us at all times to let worker nodes communicate with each other and pass information to the user. Computational cores are a key factor to the user because it tells them how many workers are available to distribute the jobs around. In this case, my computational core is 4 which means that the computer can generate only 4 workers. The methods that were used during this lab was downloading the right configurations for matlab, which was the parallel computing extensions. This will create parallelism for the user to distribute jobs between number of workers. We use the basic functions from matlab to create algorithms such as the Euclidean algorithm to measure the distance between two vectors. As a class, we generated new scripts within the editor to run programs that will generate the times to see which amount of worker nodes is the fastest. In certain parts of the lab, we created scripts for the loni cluster to generate a certain number of workers to do the job and create the time to how long each number of workers took.

A. Lab1A:
    a. Line 1: create a function that will return a single value, in this case would be t
    b. Line 2: return the number of declared inputs for the function

c. Line 7-9: declare the variables
d. Line 12: Starts the timer
e. Line 14: number of points within the for loops
f. Line 16: number of dimensions within the for loops
g. Line 18: The Euclidean algorithm
h. Line 21: square the d(i) variable that completed by line 18
i. Line 24: End of the timer

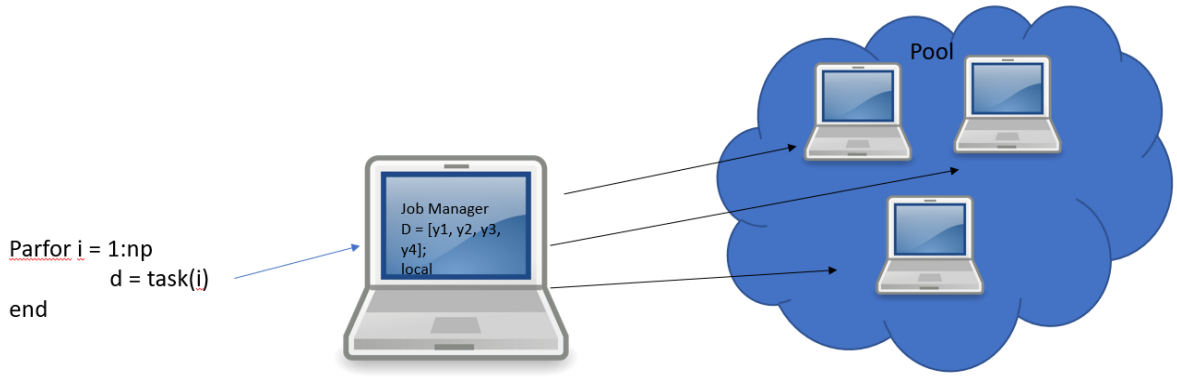
B. Lab1B:
   a. Line 1: create a function that will return a single value, in this case would be t
   b. Line 2: return the number of declared inputs for the function
   c. Line 3-8: Create a pool if it is empty to the specific number of workers.
   d. Line 13- 18: Taking something from the job manager. See a parallel task to split up, distribute the task to the number of machines in the pool, and return the result to the job manager.
   e. Line 19: Give the timer back
   f. Line 20: delete hp
C. Lab1C:
   a. Line 1: A loni runtime script
   b. Line 2-5: A for loop iterating 1-20 workers calling the LAB1B program.
   c. Then displays the run time for each of the workers
D. Lab1D:
   a. Line 1: Declaring the variables for the number of points, dimensions, and workers.
   b. Line 3 & 21: Declaring the timer
   c. Line 4: MATLAB executes the code within the SPMD body denoted by STATEMENTS on several MATLAB workers simultaneously.
   d. Line 5,6: Declaring variables to several workers
   e. Line 10-29: Running a nested for loop to conduct the Euclidean algorithm
   f. Line 17: Global Concatenation then display the result to all the workers
   g. Line 29: The client calls the index of the worker and getting the result
E. Lab1E:
   a. Line 3: Declare the number of points, and the dimensions
   b. Line 4-5: Create a pool if it is empty to the specific number of workers.
   c. Line 6: grab a random number between the number of points and the dimensions
   d. Line 7: Distributed array used the combined memory of multiple workers in a parallel pool to store the elements of an array.

e. Matrix of zeros of numbers of pool by 1
f. Line 10: Do the algorithm
g. Line 11: Returns an array in the local workspace
h. Line 12: Stop the timer

F. Lab1F:

a. Line 3: Declare the variables of the number of points and the dimensions
b. Line 5-6: Create a pool if it is empty to the specific number of workers.
c. Line 10: start the timer
d. Line 11: Taking something from the job manager. See a parallel task to split up, distribute the task to the number of machines in the pool, and return the result to the job manager.
e. Line 13-29: A bunch of if conditions and each if conditions returns a value and sent to the next lab index and being received from the past condition.
f. Line 34: Stop the timer
g. Line 39: Gets the result from the worker by calling their index and storing it in to a variable

Parfor Diagram:



III. Experiments:

Test Matrix

Labs	Description	Tools
LAB1A	This lab will use a for loop to find the Euclidean distance between two vectors. Then we gather the times and report and plot them.	HW: Local Machine SW: Matlab with parallel computing extensions Program Language: MatLab

John Do
CSC 452-001
Dr. Droz
September 25, 2019

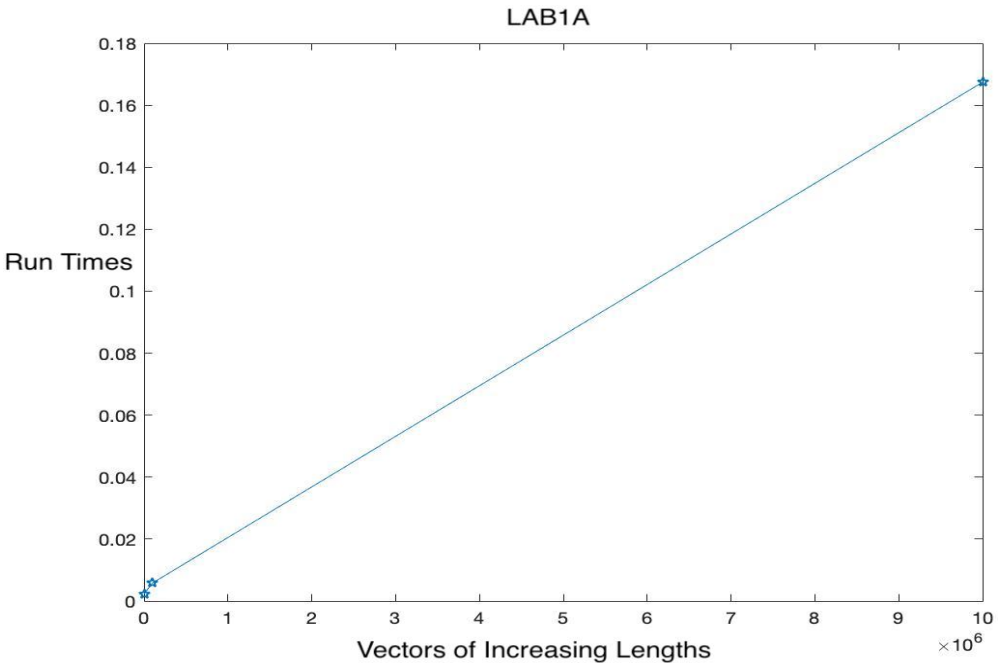
LAB1B	Like in LAB1A, the code was updated to parfor function to enact parallelisms. We use 1-8 workers on our local machine to report the run times.	HW: Local Machine SW: Matlab with parallel computing extensions Program Language: MatLab
LAB1C	This lab will analyze the run time of our Matlab program on the LONI cluster and determine what the optimal thread is.	HW: Loni Cluster SW Mathlab with parallel computing extensions; Terminal command such as **qsub** Program Language: MatLab
LAB1D	Update LAB1B code to use the spmd keyword to enact parallelism.	HW: Local Machine SW: MatLab with parallel computing extensions Program Language: MatLab
LAB1E	The code from LAB1C to use the distributed data type to enact parallelism.	HW: Loni Cluster SW: MatLab with parallel computing extensions; Terminal command such as **qsub** Programming Language: MatLab
LAB1F	Calculating distance using 4 workers in a pipelined/systolic manner by implementing ring communications	HW: Local Machine SW: MatLab with parallel computing extensions Program Language: MatLab

A. The utility that was provided is called a “loni cluster”. This cluster is located at LSU. We acquire access to it by requiring permissions from our instructor to grant us access for the loni with some allocations. After the permission was granted, we go on our mobaxterm or xquartz (Mac Users) and login with our credential with x11 forwarding in order to display certain things. Once in the loni cluster, we are allowed to run simple experiments with their resources. To run simple experiments on it, we created a scripted such as in LAB1C to generate a certain number of workers to do some job function that was created in Matlab. The program that was called by the script, it was transferred over to loni by using a software that can use SFTP, which is a secure file

transfer protocol. After the workers are done, it will generate a time which will determine the results of the experiments.

IV. Results:

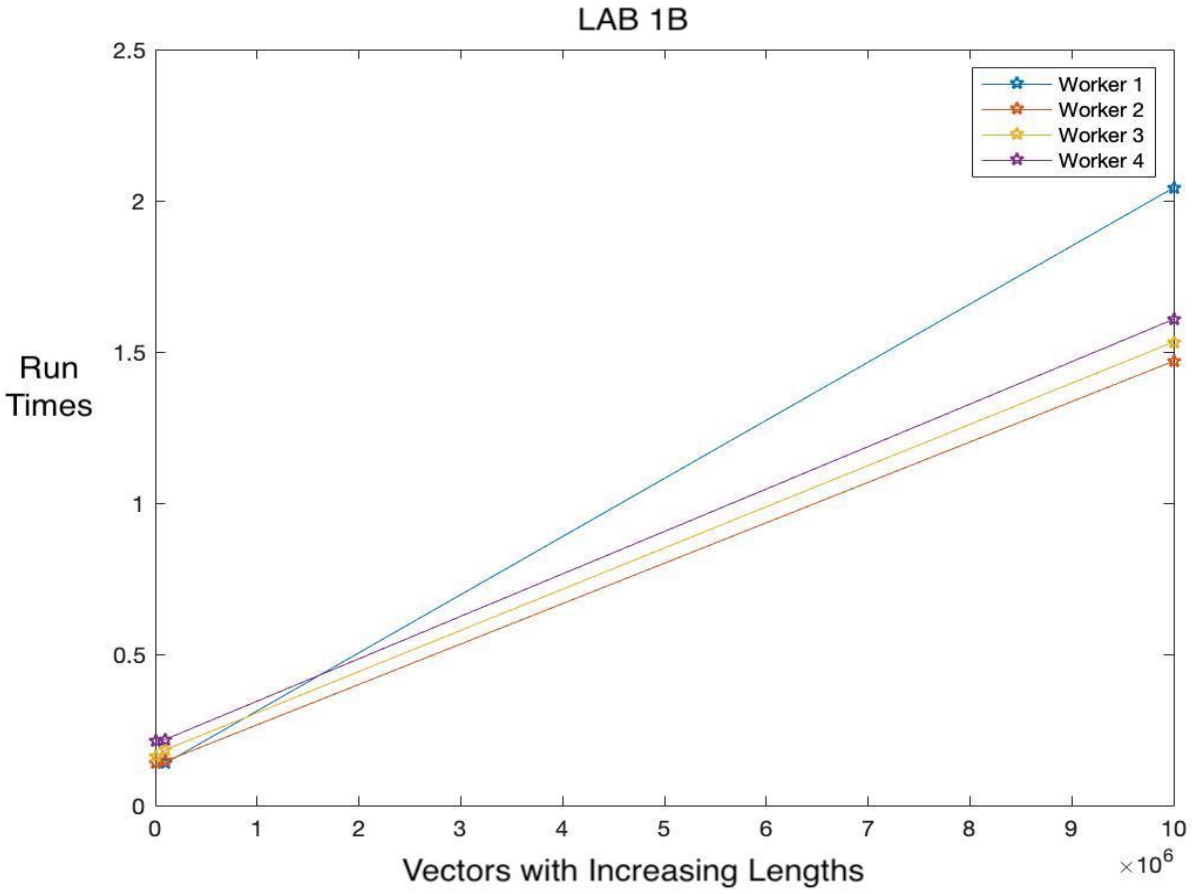
Lab1A:



Vectors of Increasing Length	Run Times
1000	0.0023116
100000	0.0058158
10000000	0.1674582

John Do
CSC 452-001
Dr. Droz
September 25, 2019

Lab1B:



		10^3		
Workers	1	2	3	4
Run Times	0.13956	0.13972	0.16154	0.21186

		10^5		
Workers	1	2	3	4
Run Times	0.13976	0.1464	0.184	0.2187

		10^7		
Workers	1	2	3	4
Run Times	2.0428	1.4694	1.5328	1.6079

John Do
CSC 452-001
Dr. Droz
September 25, 2019

LAB1F:

Vectors of Increasing Lengths	Run Times
10^3	0.17324
10^5	0.07876
10^7	1.5332

V. Conclusions & Future Work:

From the result being shown, each experiment is using the Euclidean algorithm. Lab1A shows the time being generated by using sequential for loop. The result from this portion indicates that it is getting slightly slower as the vector are increasing lengths. Lab1B is using a parfor loop, which means the iterations of statements can execute on a parallel pool of workers on your multi-core computer or computer cluster. Since my computer have 4 computation cores, the experiment can only use 4 workers. The table shows the different times comparing between each number of workers. It is acquiring the same data values as Lab1A, but it is using different number of workers to produce the output. Two number of workers would be the best use case because of less overhead to manage and it produces the fastest time in each increasing lengths of the vector. LAB1F is implementing a ring communication, which is done by identifying the lab index and running the algorithm and sending the result to the next index. This would be the favor of doing simple algorithm because it creates a cleaner way of producing data and sending it to the next lab index. In lab1B, it shows that 4 workers are running the algorithm with the vector increasing lengths, but it is slower than Lab1F with the same number of workers. In future work, this will help us program algorithms in Matlab and learn how input data values are being transmitted and how the job is being distributed between different number of workers. In the future, we would know what an optimal amount of threads and workers would be to help process the job faster and does not require a lot of overhead to manage the workers. This will be beneficial on working with complex algorithms that require larger number of data being inputted and visualizing how the data is going to be outputted. This will help us during future tasks of using the loni cluster because of parallelism between number of workers that is provided.

GitHub Link: https://github.com/jonathanmdo/CSC_452_Labs/tree/Lab1