In this project you are going to implement a parallel algorithm with C/C++ language using MPI library.

MapReduce

Everyday we create tremendous amounts of data in our every activity. Twitter adds 400 million tweets to its database in every day. The sensors in the Large Hadron Collider at Cern records petabytes of data each year. We can find many more examples from astronomy, biology, internet activities, sensor networks, etc. This makes big-data processing, today’s one of the biggest computer science and engineering problems.

Distributed processing is the general approach for handling large volume data, but designing an efficient distributed system is a challenging task. There are some general distributed programming frameworks in order to simplify the implementation. One of them is the MapReduce model. There are many libraries for MapReduce, so that the programmer does not care about the distribution of the data, instead he or she supplies the necessary map and reduce functions. However, we are not going to use any library; instead, we borrow the idea from this model, and implement our solution using MPI library in C or C++.

In this project, you are going to demonstrate a small distributed data processing solution using the MapReduce programming model. This model consists of map and reduce steps. In the map step, master node takes the input, divides it and distributes to worker nodes and each worker node works on its own data independently. In the reduce step, the master node collects the answers from the workers, and combines them to generate the final result. (This programming model can be implemented in a multi-level way, i.e a worker node can map its input to other idle workers and collect the results, but we are not going to implement this.)

Problem Definition

You are going to extract records and calculate statistics from a large gene expression database. The data set consists of 2467 genes. Each gene can belong to one of the following 6 classes: tricarboxylic acid cycle (TCA), respiration (Resp), cytoplasmic ribosomes (Rib), proteasome (Protes), histones (Hist) and helix-turn-helix proteins (HTH). There are also 79 expressions for each gene, corresponding to different measurements.

The data is stored in a tab separated file where the first column is the unique identifier of the gene (ORF=open reading frame), the second column is the name, next 6 columns are the class labels and the remaining 79 columns are the measurements (Table 1).

When your program starts, the master node should load the data, divide and distribute it among the worker processors. Then, the master node should wait for the user to input a query.
Table 1: Gene Expressions. If a gene is labeled with a class its corresponding value is 1, otherwise it is -1.

There will be 2 types of queries as listed below. When your program answers a query, it should not terminate, instead wait for the next query. Your program should terminate when the user enters:

`quit`

1. Finding a record

The user may want to see the data about a single gene. For example, if the user wants to see the gene YMR056C, he or she will enter:

`gene YMR056C`

Your output should contain all information about the gene.

The output format is as follows:

```
YMR056C
Name: AAC1 TRANSPORT MITOCHONDRIAL ADP/ATP TRANSLOCATOR
TCA: -1
Resp: -1
Ribo: -1
Proteas: -1
Hist: -1
HTH: -1
alpha 0: -0.18
alpha 7: -0.58
... ...
```

2. Calculating Statistics

The user may wonder the mean and the standard deviation of the measurements of genes belonging to a specific class. For example, if the user wants to list the statistics for the TCA class, he or she enters:

`class TCA`

You have to output the mean and standard deviation of the 79 measurements of the genes labeled as TCA:

```
Mean     Std
-0.0059  0.3435
```
Important Points

- The data set is taken from http://compbio.soe.ucsc.edu/genex/expressdata.html
- There are some missing measurement values in the gene expressions. You need to replace them with zeros.
- The master node is supposed to distribute all of the data and control the process. In your design, it is not allowed to process data, instead it will only collect and combine the workers answers.
- Linear search is prohibited, in the first part. Remember that this small data set is actually a demonstration for a really big data set. Therefore, do something more clever like sorting the genes according to orf values. And of course, do not sort the whole data set in the master node.
- For the second part, try to send as less data as possible between master and workers. For example, sending all measurements to master processor and letting it do all the calculations is not appropriate.
- Be sure to test your design with different number of workers processors.
- If you choose C++ for programming, you still need to use C MPI bindings since C++ MPI bindings depreciated.
- See http://www.cmpe.boun.edu.tr/gungort/informationsstudents.htm for general information (submission process, documentation, etc.) about programming projects.

Appendix

Let $X = \{x_1, x_2, \ldots, x_N\}$ be a set of $N$ measurements. We calculate the mean of this set as

$$\bar{X} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

The variance of the data set is calculated as

$$s^2 = \frac{1}{N - 1} \sum_{i=1}^{N} (x_i - \bar{X})^2$$

We call $s$ the standard deviation.