HW-OpenMP-k-Means

In this homework, you are required to use MPI to redo your homework that using k-means algorithm for the image segmentation.

First let's review the k-means algorithm:

Step 1: **Initialization**. Read an image from a JPG file. Choose k pixels from the image. (You can label all the pixels from 1 to N, and choose one pixel every N/k pixel). The colors of the k pixels are called generators. So now we have k generators, labeled from 0 to k-1. (Remember that a generator is a color, not a pixel.)

Step 2: **Grouping the pixels**. Thinking each color as a vector composed by (R, G, B). Then we can define the distance between every two colors. For each pixel, find the generator which has the shortest color distance to the pixel's color. If the index of the generator is i, then put the pixel to group i. After all the pixels are grouped, we get k groups of pixels.

Step 3: **Get new generators**. For each group of the pixels, find the average color of the group. (Average on R, G, B.) And set the average color as the new generator. And finally, we get k new generators.

Step 4: **Jump back to Step 2**. Step 2 and Step 3 form an iteration. Repeat the iteration for a predefined number, for example, 20 times. And then go to Step 5.

Step 5: **Finishing**. For each pixel, replace its color by its group's generator. And write the new image into a JPG file.

Second, we need consider about how to apply MPI towards this algorithm. Step 1 can be done by the Master process. We mainly need helps from the Workers for the Step 2 and Step 3. And to minimize the communications, you can think about combine these two steps together:

From old set generators to new set of generators.

The pseudo code combing Step 2 and Step 3 is as following:

```
Initilize sum[i] = {0,0, 0} (for R, G, B), count[i] = 0 for i = 0, …, k
Loop over all pixels
        For each pixel p,
                Find out the closest old generator, say s;
                sum[s] += color of p;
                count[i] ++;
        endfor
end loop
For i = 0, …, k
        generator[i] = sum[i] / count[i];

Endfor
```

Remember that here, generator[i] and sum[i] are colors containing R, G, B.

Now, you can think about how to apply MPI towards this k-means algorithm. Below is some hints for you:

1. In Step1, Master reads the JPG file, and broadcast number of pixels to all other processes. And all processes allocate the memory for holding the pixels they are going to do the segmentation. And use MPI_Scatter function to scatter the picture information to all processes. (Just as what we did in gray color imaging homework)
2. Master initialize the generators, and broadcast to all other Workers.
3. Step2 and Step3 is combined and described in above pseudo code. Each process get its own sum[i] and count[i]. And use MPI_Reduce function to summarize into the Master. The Master calculates the new set of generators. And broadcast to all other Workers.
4. In Step 5. Every process updates the pixels that it is responsible of. Use MPI_Gather to collect the pixel information to Master. Master write the final image into the output JPG file.

Test your code with different processes. Timing your code.

Write a final report and submit together with your code.