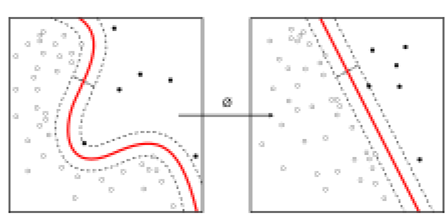
K-Means Parallel Computing Documentation

Lecturer: Dr. Boris Morose

Student: Orel Damari
Student ID: 204739825

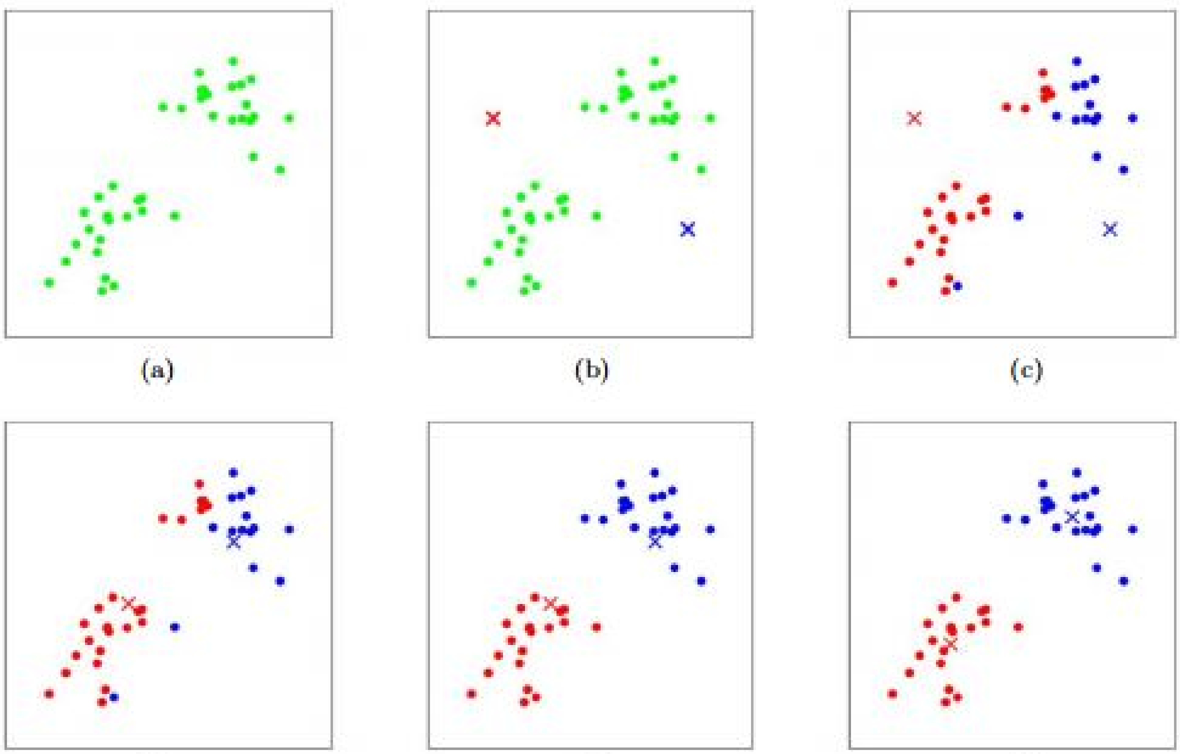


K-Means Algorithm:

k-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.

In this project I Implemented the K-Means in the following steps:

1. Calculating number of points
2. Calculate distances
3. Calculate clusters
4. Move points according to Velocity
5. If Quality Measure not enough and max time did not achieved : Back to step 2



(a) (b) (c)

(d) (e) (f)

What Parallelized?

MPI (Message Passing Interface):

Sharing & dividing the points between the processes.

Cuda:

Moving the points
Calculating distances
Calculating clusters
Classifying point to cluster

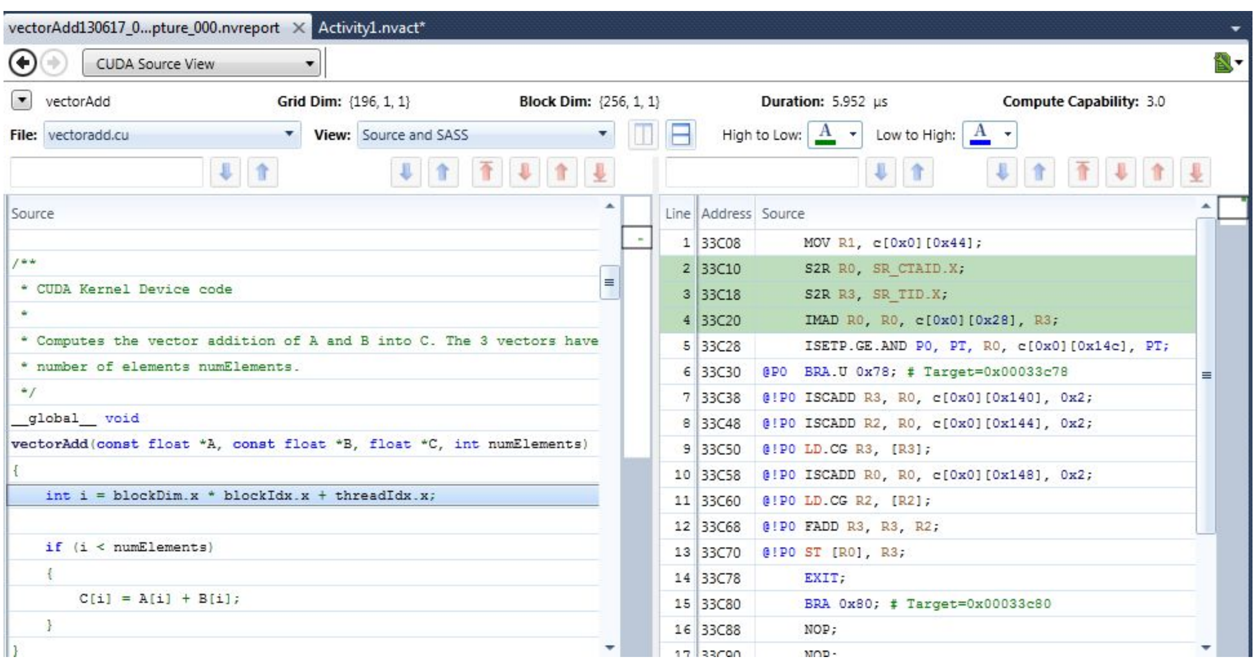
OpenMP:

Quality Calculations

IMPLEMENTATION:

I Implemented the K-Means in the following way:

1. Read file, number of points, Clusters, Quality Measure, Time, dT.
2. Calculate number of points for every process. If there is a remainder it stays at the Master process.
3. Using MPI - send array of points to the processes.
4. Every process know which points from this array to read- according to the MPI process id.
5. Every process calculate the number of blocks needed in the cuda- According to the check of how many max threads can be in every block.
6. The Cuda move the points- One gpu thread to each point.
7. Calculate distances and new Cluster- using cuda.
8. Returning an array that classifies every point to the right Cluster.
9. Using OMP - Calculate the Quality Measure.
10. Check if Maximum time iteration achieved or Quality Measure achieved.
11. If Yes- Finish and write to file the clusters.
12. If No- go back to step 3.



Formulas:

Quality Measure: (for k=3)
q = (d1/D12 + d1/D13 + d2/D21 + d2/D23 + d3/D31 + d3/D32) / 6

Distance between two points:
D(X1-X2) = sqrt(X2^2 - X1^2)

Points Movement:

$x_i(t) = x_i + t*v_{xi}$

$y_i(t) = y_i + t*v_{yi}$

KMeans:

Given a set of observations (x_1, x_2, …, x_n), where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into k (≤ n) sets S = {S_1, S_2, …, S_k} so as to minimize the within-cluster sum of squares (WCSS) (i.e. variance). Formally, the objective is to find:

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 = \underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i$$

where $\boldsymbol{\mu}_i$ is the mean of points in S_i. This is equivalent to minimizing the pairwise squared deviations of points in the same cluster:

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} \frac{1}{2|S_i|} \sum_{\mathbf{x},\mathbf{y} \in S_i} \|\mathbf{x} - \mathbf{y}\|^2$$

The Equivalence can be deduced from identity

$$\sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 = \sum_{\mathbf{x} \neq \mathbf{y} \in S_i} (\mathbf{x} - \boldsymbol{\mu}_i)(\boldsymbol{\mu}_i - \mathbf{y})$$

Because the total variance is constant, this is also equivalent to maximizing the squared deviations between points in *different* clusters (between-cluster sum of squares, BCSS).