**Parallel implementation of K-Means**

Final project

Course 10324, Parallel and Distributed Computation

2017 FALL Semester

# Problem Definition

Given a set of points in **2**-dimensional space. Initial position **(xi, yi)** and velocity **(vxi, vyi)** are known for each point **Pi**. Its position at the given time **t** can be calculated as follows:

xi(t) = xi +t\*vxi

yi(t) = yi +t\*vyi

Implement simplified K-Means algorithm to find **K** clusters. Find a first occurrence during given time interval [0, T] when a system of **K** clusters hasa Quality Measure **q** that is less than given value **QM**.

**Simplified K-Means algorithm**

1. Choose first **K** points as a cluster centers.
2. Group points around the given cluster centers - for each point define a center that is most close to the point.
3. Recalculate the cluster centers (average of all points in the cluster)
4. Check the termination condition – no points move to other clusters or maximum iteration LIMIT was made.
5. Repeat from 2 till the termination condition fulfills.
6. Evaluate the Quality of the clusters found. The Quality is equal to an average of diameters of the cluster divided by distance to other clusters. For example, in case of k = 3 the quality is equal

**q = (d1/D12 + d1/D13 + d2/D21 + d2/D23 + d3/D31 + d3/D32) / 6**,

where di is a diameter of cluster **i** and Dij is a distance between centers of cluster **i** and cluster **j**.

Input data and Output Result of the project

You will be supplied with the following data

* **N** - number of points
* **K** - number of clusters to find
* **LIMIT** – the maximum number of iterations for K-MEAN algorithm.
* **QM** – quality measure to stop
* **T** – defines the end of time interval [0, T]
* **dT** – defines moments t = n\*dT, n = { 0, 1, 2, … , T/dT} for which calculate the clusters and the quality
* Coordinates and Velocities of all points

**Input File format**

The first line of the file contains **N K T dT LIMIT QM**. Next lines are Initial Positions and Velocities of the points

For example:

5000 4 30 0.1 2000 7.3

2.3 4. 5 6. 55 -2.3

76.2 -3.56 50.0 12

…

45.23 20 -167.1 98

**Output File format**

The output file contains information on the found clusters with the moment when the Quality Measure QM is reached for first time. For example:

**First occurrence at t = 24.5 with q = 6.9**

**Centers of the clusters:**

1.123 34

-5.3 17.01

33.56 -23

14.1 98

# Requirements

* Implement the K-MEANS algorithm explained in the class (see above). Use first K points at t=0 as initial positions of the centers of the clusters. In case that in some iteration there will be no points in cluster – keep its center for the next iteration.
* The input file **input.txt** initially is known for one machine only. The results must be written to the file **output.txt** on the same machine.
* Calculations stops after the Quality Measure was reached.
* Diameter of cluster is the largest distance between two points of this cluster.
* Distance between two clusters is a distance between centers of these clusters.
* The computation time of the parallel program must be faster than the one for sequential solution.
* Be ready to demonstrate your solution running on at least three computers (if MPI is used)
* **No code sharing between students is allowed.** Each part of code, if any, which was incorporated to your project must be referenced according to the academic rules.
* Be able to explain each line of the project code, including those that was reused from any source.
* The set contains at least **10000** but not more than **3000000** points.

**Grade Policy**

* **60 points** for the effective **proper** parallel implementation of the problem with two components: ***MPI+OpenMP*** or ***OpenMP+ CUDA*** or ***MPI+CUDA***. The project that produce wrong results will not be accepted.
* **10 points** for implementation in ***MPI+OpenMP+CUDA*** configuration.
* **10 points** for the documentation of your solution – clear explanation what and how the problem was parallelized, what is a rational of choosing the specific architecture, complexity evaluation.
* **10 points** for the code quality – modularity, generality, self-explanatory, organization.
* **10 points** for the Load Balancing.

***Additional Bonus for the project grade***

**5 points** for implementation under LINUX OS

**5 points** for implementation with OpenCL

**5 points** for implementation of sophisticated variation of the K-MEANS algorithm(must be approved by lecturer).

**5 points** for your own proposal (must be approved by lecturer).

# הפרויקט יוגדר כמטלת הקורס. הגשת התוכנה והתיעוד רק דרך מערכת Moodle לאחר ההגנה.

# יישום והגשת הפרויקט ביחידים בלבד.

בהצלחה