Parallel implementation of Binary Classification

Final project

Course 10324, Parallel and Distributed Computation

2019 Summer Semester

Program expects an input file including classifier info, N points with K dimensions and a sign.

Program will output a file that contains minimum time t and classifier quality q plus separating plane weights, if found.

Problem parallelization rational:

Each process will receive classifier info and all points found in file from MASTER.
Data sent using custom **MPI_TYPES** and **MPI_PACK**.

All times **'t'** up to **'tlimit'** in **'dt'** intervals will be divided between available processes.
E.g. iteration 1: (p0, t0) (p1, t1). Iteration 2: (p0, t2) (p1, t3) …
at each iteration's end MASTER will receive **'q'** – classifier quality information from each process and will decide if another iteration is needed or if accepted quality is found.
Communication was implemented using **MPI**.

Iteration procedure:

Each process will set its points location according to current time **'t'** using **CUDA** .
CUDA was chosen since the points locations are independent from one another.

Each process will run the **'classify'** function to iterate over all of the points while fixing the weights, if needed. Number of iterations are as defined in the classifier.
Sequential implementation was chosen for this part since the parallel implementation required too much communication that caused major slowdowns.
Unused parallel implementations using OpenMP or CUDA can be found in the code.

Each process will then count the number of misclassified points using **OpenMP**, including parallel for and reduction.
CUDA implementation was considered for this part, but OpenMP was found to be faster.
Unused sequential and CUDA implementations for this part can be found in the code.

Complexity evaluation:

Each process Points were allocated sequentially and then assigned to pointers for faster caching considerations. This decision also made CUDA memory allocation easier.
CUDA memory was allocated once at the beginning of each process initialization and memory was reused for Points advancement.