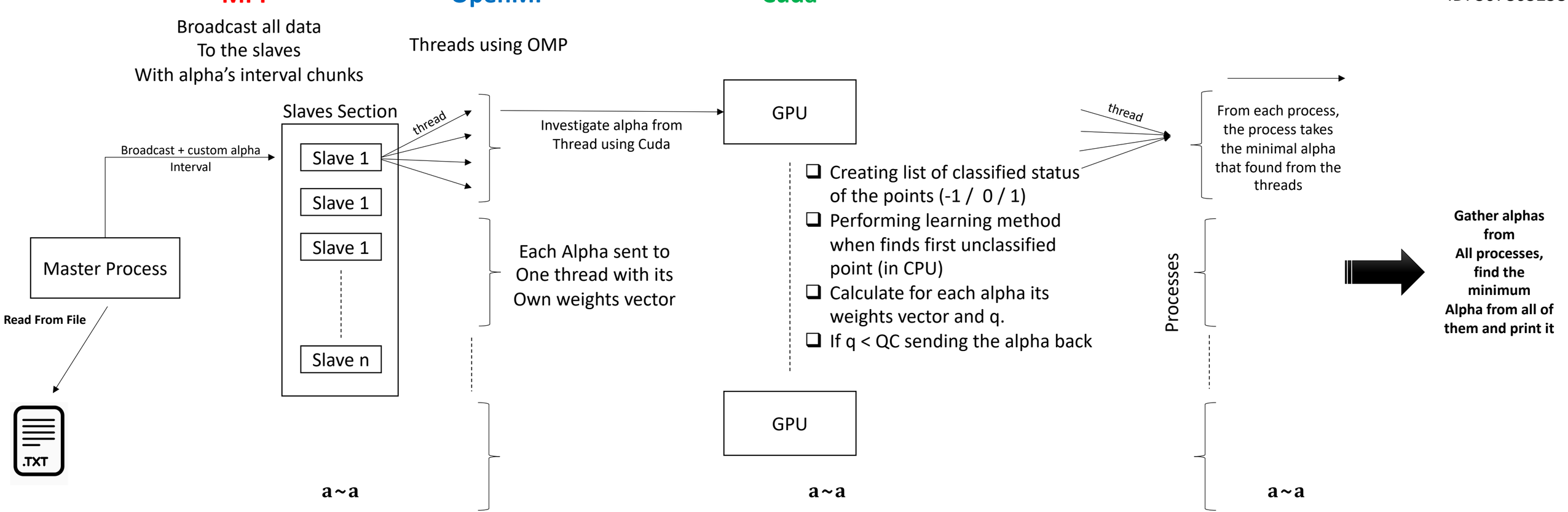
Parallel implementation of Binary Classification - Project explanation

By: Giron Aptik
ID: 307863258



Parallel implementation of Binary Classification - Project explanation

By: Giron Aptik
ID: 307863258

Master Process read the data from file.
Broadcast message with all data performed.

- MPI divides the alphas interval to sections and send each section (alpha, alphZero, alphaMax) to one process (Master & Slaves). **(Send Job a~a)** (Step 1,2, 9, 10)

- OpenMp divides each interval of alphas for each process to one alpha per thread. **(Send Job a~a)**

- In each thread, the program investigate each alpha with Cuda. Perceptron with initiated weight vector, specific alpha and q sending TO function called “InvestigateAlpha”.

  Cuda Work: (Step 2-6)

  Sign of discriminant function for each point.
  Find the first unclassified point and updating the weights vector accordingly.
  Calculate the nMis and the q after limit iteration or if all points already classified.

  Cuda returns each perceptron with q (if q < QC) and its weights vector.

  (Work on each alpha per job - a~a)

- OMP return the minimum alpha for each thread we found. (Step 7-8)

- Gather function performed to locate all the perceptron that calculated from all process (Master + Slave).

The master process searching for the minimum alpha the we calculated,

The master prints the results to the Output file.

I chose the specific architecture to make sure that the program uses all the parallel tools at every step that it's possible. The only time that the program uses only one tool (MPI) is at the start and the end when process MASTER read the data, gather and prints the results.