Efficient k-Means on GPUs

Clemens Lutz[1], Sebastian Breß[1,2], Tilmann Rabl[1,2], Steffen Zeuch[1], Volker Markl[1,2]

German Research Center for Artificial Intelligence

[1]firstname.lastname@dfki.de
[2]firstname.lastname@tu-berlin.de

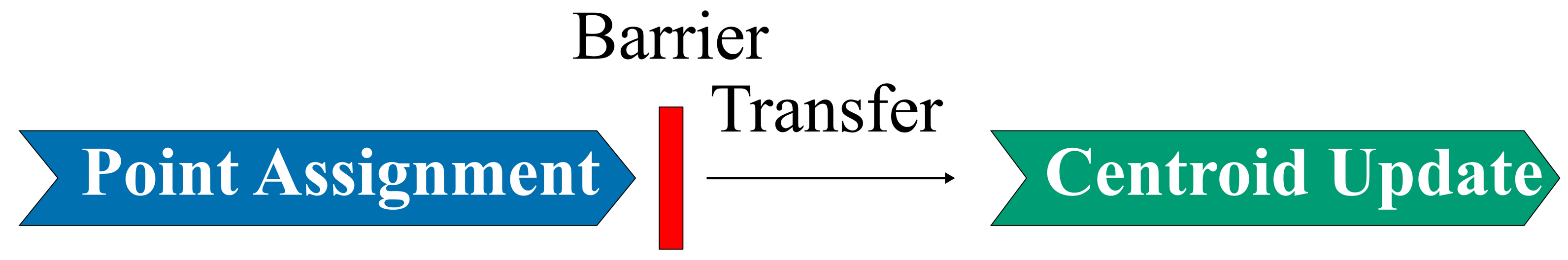
Abstract

k-Means is widely used in diverse fields of study. Quick execution allows practitioners to explore more data.

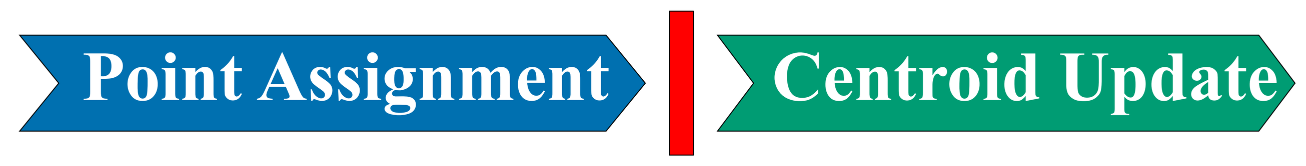
Fast, data-parallel GPUs expose the cross-processing problem and the multi-pass problem as bottlenecks.

We present a new centroid update algorithm for GPUs & fuse GPU kernels for a single data pass per iteration.

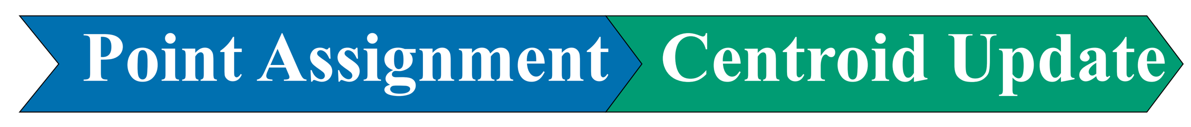
Problem 1: Cross-Processing between GPU and CPU



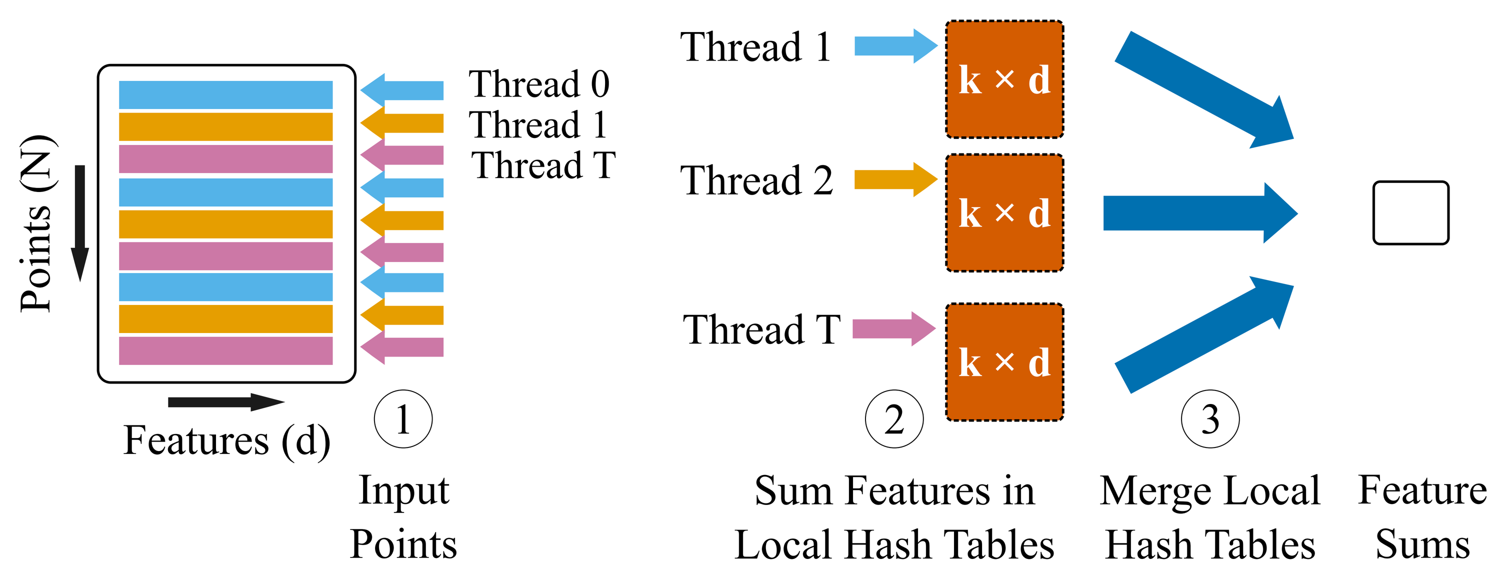
Problem 2: Multiple Data Passes on same Processor



Goal: Single Data Pass



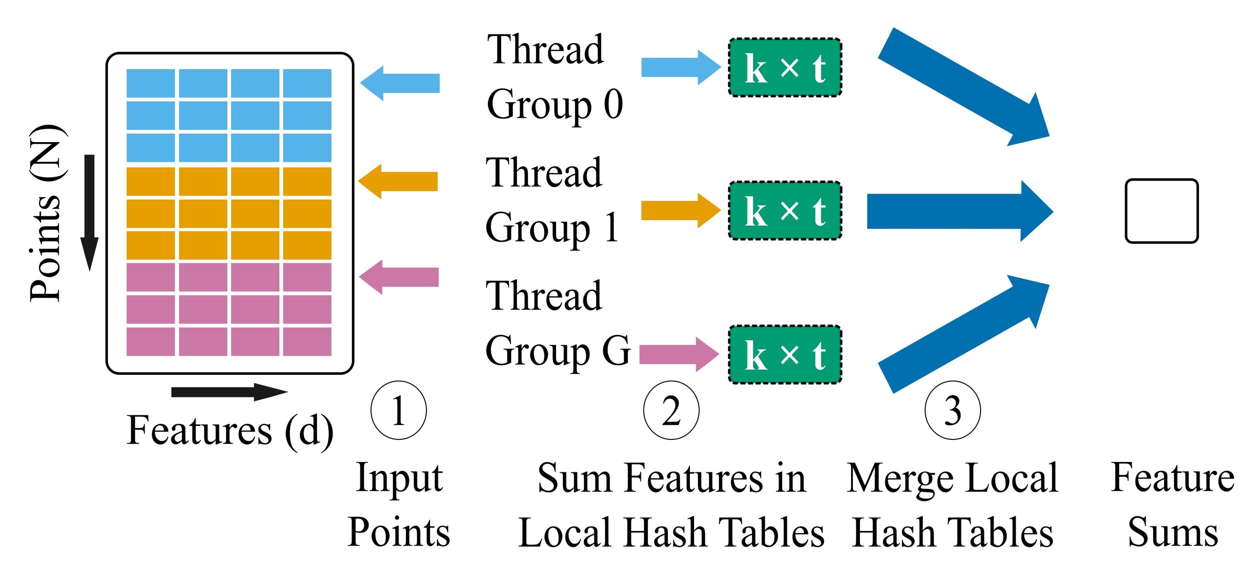
Thread-wise Centroid Update



Partitioning on data points requires each thread to store all data features in cache.

1. Solve Cross-Processing

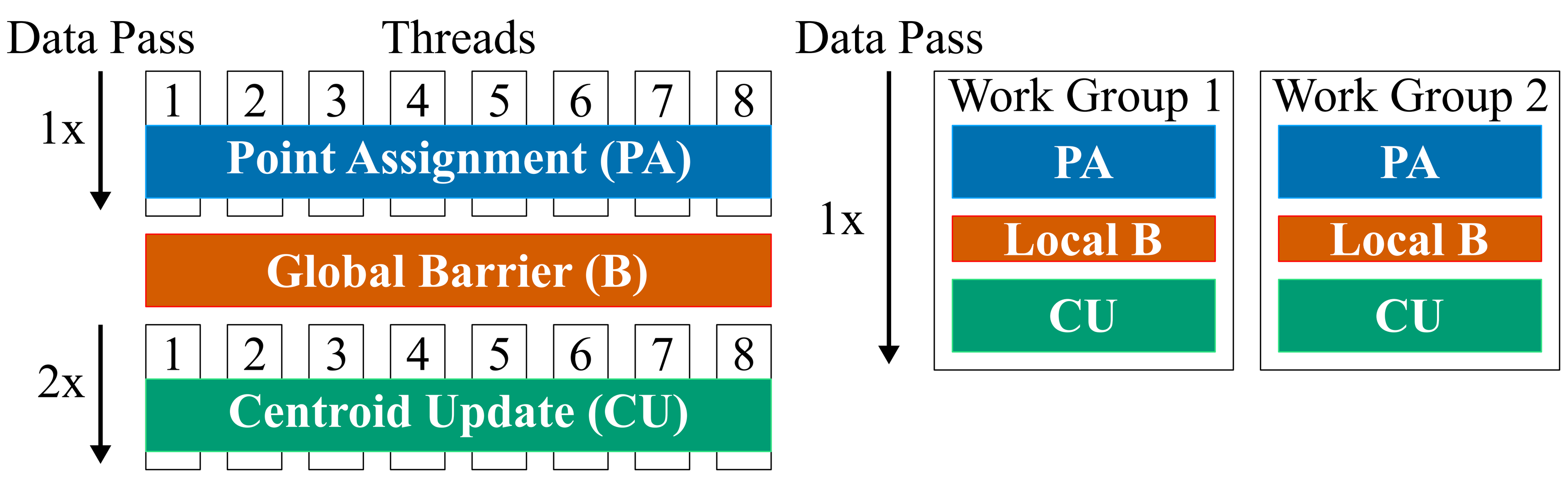
Thread-Group Centroid Update



Partitioning on points and features unties cache footprint from number of features.

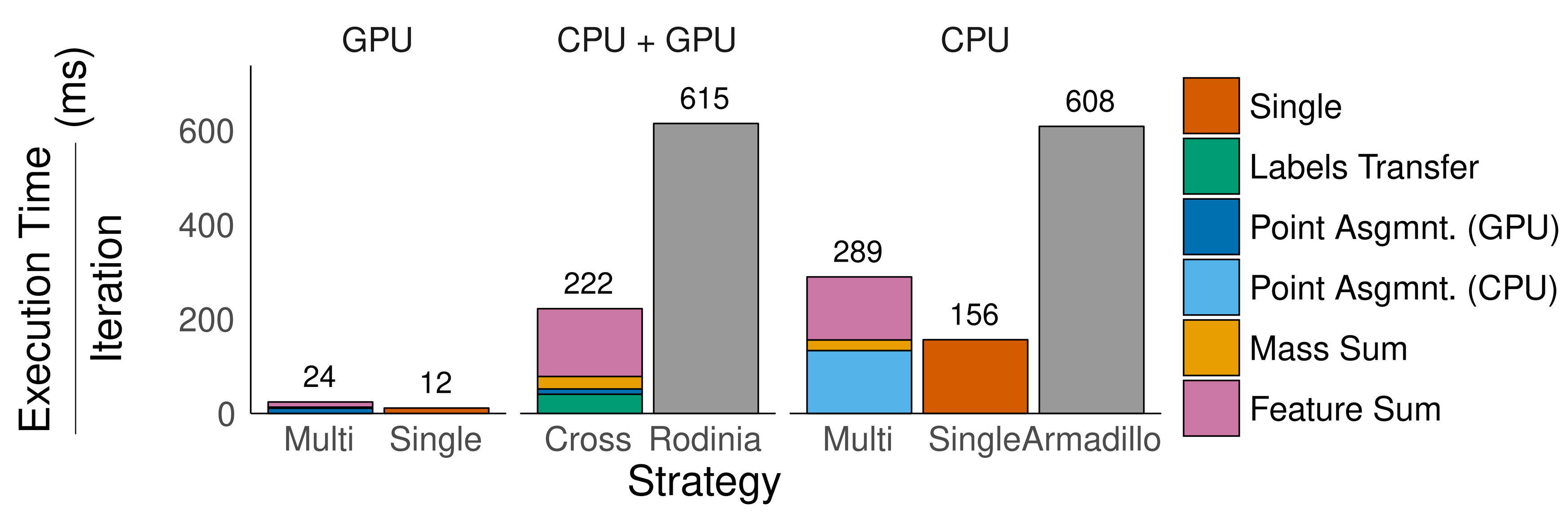
2. Solve Multi-Pass

Thread-Group-Local Synchronization



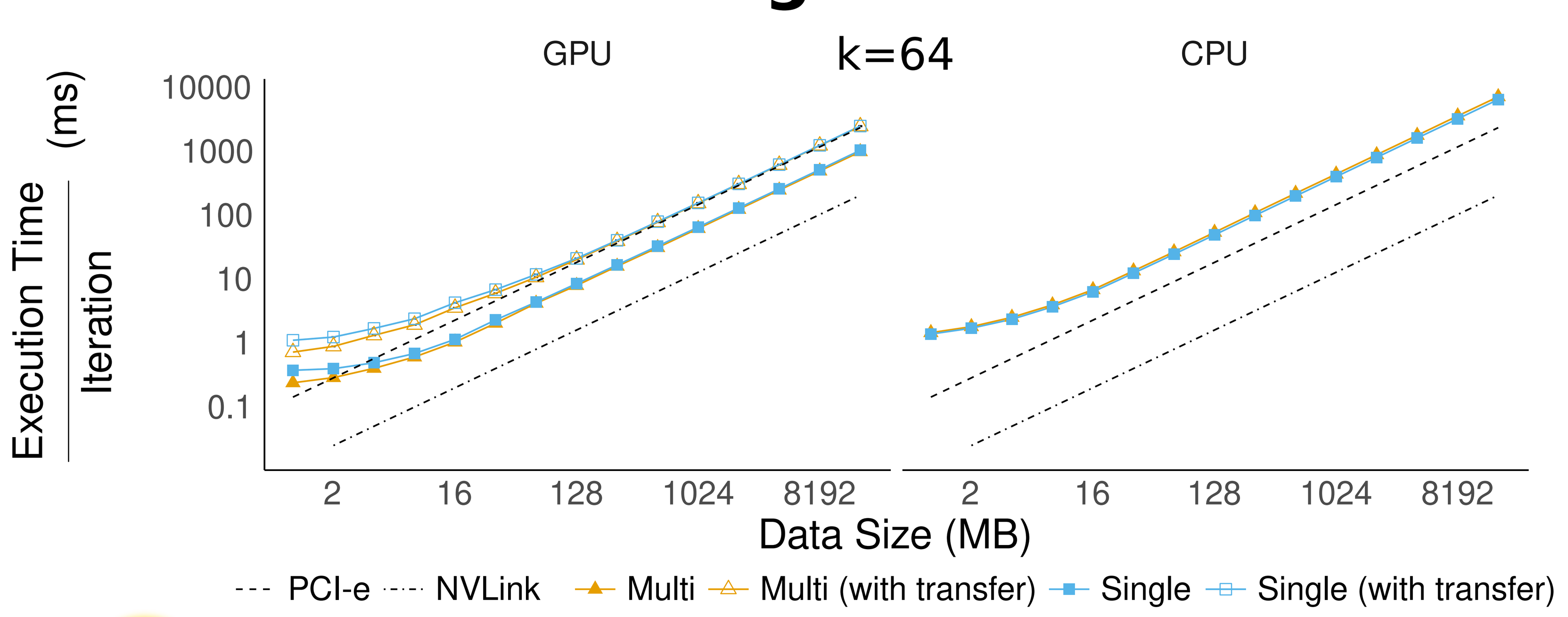
Multi-Pass Strategy vs. Single-Pass Strategy

Execution Strategy Comparison



Cross-processing incurs transfer overhead. Single-pass reduces data accesses by half.

Scale to Large Data Sets



GPU clusters faster than CPU despite data transfer on each iteration.

Take Home

Partitioning by both points and features reduces cache footprint of centroid update for up to 10× faster execution on GPUs.

A single data pass increases throughput by up to 2×, but enlarges cache footprint.

20× better overall performance paves the way for high-bandwidth NVLink.

Open Source Repository

github.com/TU-Berlin-DIMA/CL-kmeans

Funding Acknowledgements

This work was funded by the EU projects SAGE (671500) and E2Data (780245), DFG Priority Program "Scalable Data Management for Future Hardware" (MA4662-5), and the German Ministry for Education and Research as BBDC (01IS14013A).