Efficient k-Means on GPUs

Clemens Lutz, Sebastian Breß, Tilmann Rabl
Steffen Zeuch, Volker Markl
k-Means State-of-the-Art

- Accelerate point assignment on GPU
  - High parallelization power

![Diagram showing the k-Means algorithm with Point Assignment, Point Labels, Centroid Update, and New Centroids]
Problem 1: Cross-Processing

- Processing split between two processors
  - PCI-e bus transfer for labels and centroids

Incurs high PCI-e transfer costs

Diagram:
- Point Labels
- Point Assignment
- PCI-e
- Centroid Update
- New Centroids
Solution: Update Centroids on GPU

- New algorithm for centroid update
  - Reduce cache footprint

Point Assignment

Centroid Update

- Untie cache footprint from data features
- Eliminate PCI-e transfer costs

Thread-wise 1D partitioning

Data Points

Features (d)

Thread 1
Thread 2
Thread T

Hash table per thread

$\text{k} \times \text{d}$

$\text{k} \times \text{d}$

$\text{k} \times \text{d}$

Work Group 2D partitioning

Data Points

Features (d)

Thread Group 1

Thread Group 2

Hash table per thread group

$\text{k} \times \text{t}$

$\text{k} \times \text{t}$
Problem 2: Multi-Pass

- Point assignment and centroid update each make a data pass
  - Global barrier necessary due to transposed data access
  - Transpose on-the-fly and use local barrier inside thread group instead

Solution: Single pass per iteration
Benefits of Centroid Update on the GPU and a Single Data Pass

**GPU:**
- Cross-processing problem: 10×
- Multi-pass problem: 2×
- Overall: 20×

**CPU:**
- Multi-pass problem: 1.8×
Contributions – “Efficient k-Means on GPUs”

Cross-Processing Problem
→ Efficient Centroid Update for GPUs

Multi-Pass Problem
→ Single-Pass k-Means

https://www.dfki.de/web/forschung/publikationen?pubid=9767