Geo-located tweets from Twitter gathered on Nov. 10, 2011

**DBSCAN**

- Aims to separate clusters from noise
- Finds arbitrarily shaped clusters
- # of clusters not needed

**DBSCAN Scaling**

- DBSCAN's serial performance limits feasible calculations to tens of millions of points
- Need parallelism to cluster datasets with billions of points

**DBSCAN example: Chicago, IL**

- \( \epsilon = 1^\circ \) lat/long
- MinPts = 400

**DBSCAN Execution**

1. Run DBSCAN (BE)
   - Run DBSCAN to classify points into clusters on local partition

2. Pick Representative Points (BE)
   - Representative points are a small finite set of points that represent all core points in a \( \epsilon \times \epsilon \) box
   - Used to detect cluster overlap
   - A core point that would cause a merge has to fall within \( \epsilon \) of a representative point

3. Merge Overlapping Clusters (CP)
   - Detect possible cluster overlap with grid
   - Check for representative point overlap
   - Calculate the diff of the non-core points
   - Calc distance from representative points
   - Merge if at least ONE’s distance from core point

4. Color the Clusters (FE)
   - Color each cluster
   - Send coloring to BE for cluster output

**Packet Contains**

- Cluster non-core points (size bounded by MinPts)
- Representative points for each \( \epsilon \times \epsilon \) box in cluster

**0. Prep Input Data**

- Divide data into \( \epsilon \times \epsilon \) boxes
- Form roughly equal partitions from boxes
- Outline each partition with "shadow area"
- Target # points per BE

**DBSCAN**

- 800K – GPU DBSCAN
- 50K – CPU DBSCAN