Fast Clustering using MapReduce

Alina Ene, Sungjin Im, Benjamin Moseley
UIUC

KDD 2011

Clustering Massive Data

- Group web pages based on their content
- Group users based on their online behavior
  - Finding communities in social networks
- The web graph has a trillion edges [Malewicz et al.]
- Sequential algorithms are unusable

MapReduce

- We work with the MapReduce model of computation of Karloff-Suri-Vassilvitskii (SODA 2010)

Introduction to MapReudce

- MapReduce runs in multiple rounds
- Each machine is `separate'
- Memory and number of machines are typically much smaller than the input data

Running Time in MapReduce

- Time is constrained by the number of rounds due to moving data
- Minimize the number of rounds
- Keep running time in each phase polynomial

MapReduce Class [Karloff et al.]

- N is the input size and ϵ is a constant
- Less than $N^{1-\epsilon}$ memory on each machine
- Less than $N^{1-\epsilon}$ machines
- Mappers/reducers are $\mathrm{poly}(N)$ computable
- $\mathcal{MRC}^0$: algorithms that run in $O(1)$ rounds

Algorithmic Design in MapReduce

- No one machine can see then entire input
- Machines are oblivious to the data on other machines, i.e. no communication between machines during a phase
- Total memory is $N^{2-2\epsilon}$

MapReduce vs. PRAM

- PRAM
  - No limit on the number of processors
  - Memory is uniformly accessible from any processor
  - No limit on the memory available in the system
- Difficult to adapt PRAM algorithms to MapReduce

Previous Work On MapReduce

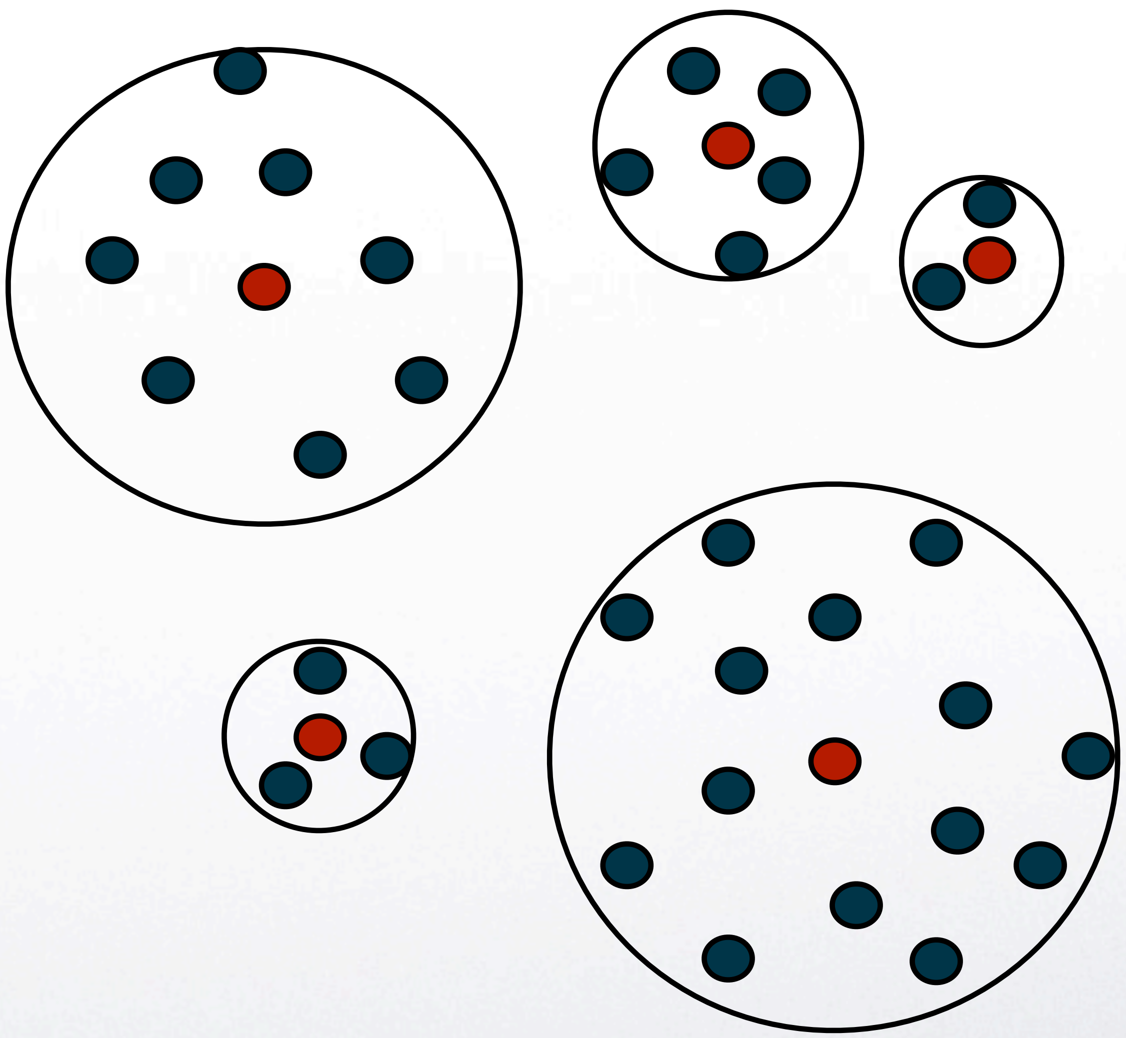
- Previous work requires $O(d)$ rounds to compute a BFS where d is the diameter of the graph [Lin and Dyer, Kang et al.]
- $O(1)$ rounds to compute a spanning tree or connected components for dense graphs [Karloff et al.]
- $(1 - \frac{1}{e} - \epsilon)$ approximate for max-cover in $polylog(n)$ rounds [Chierichetti et al]
- $O(1)$ round algorithms for maximal matching, minimum cut, edge cover and vertex cover in dense graphs [Lattanzi et al.]

Our Contributions

- We give constant factor approximation algorithms that are $\mathcal{MRC}^0$
  - We consider kCenter and kMedian
- Empirical evaluation
- We focus on kMedian in this talk

Clustering

- Input: n points in a metric space, together with pairwise distances between them
  - Input size $N = \tilde{\Theta}(n^2)$
- Output: a subset C of k points, and an assignment $f : V \to C$
  - This talk: k is a const.



kMedian Clustering

- Minimize the total distance to the centers

$$\min_C \sum_{v \in V} d(v, C)$$

- Weighted version

$$\min_C \sum_{v \in V} w(v) \cdot d(v, C)$$

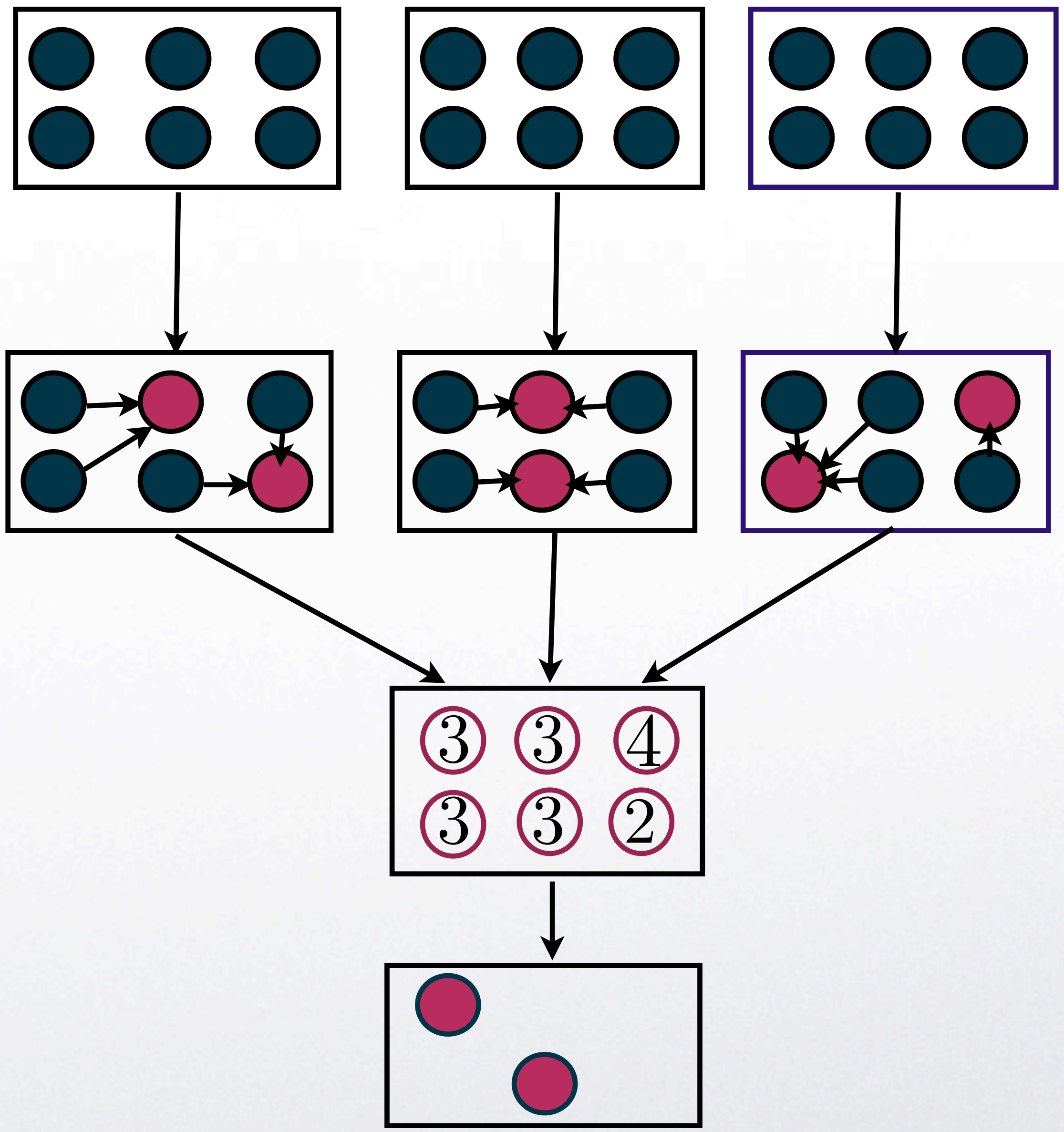
- Sequential algorithms
    - $3 + 2/c$ in $O(n^c)$ time [AryaGKMMP 01]
    - MAX SNP-hard [GuhaK 98]

Algorithms

- Two algorithms
  - Partition-based algorithm
  - Sampling-based algorithm

Partition Algorithm

- Partition the points into blocks of the same size $\sqrt{n}$
- Find k centers from each block
- Cluster the centers



Analysis

- Constant factor approximation
  - 3α approximation
- Constant number of rounds
- Memory: $\Theta(n) = \Theta(\sqrt{N})$
- Machines: $\Theta(\sqrt{n}) = \Theta(N^{\frac{1}{4}})$
- In $\mathcal{MRC}^0$

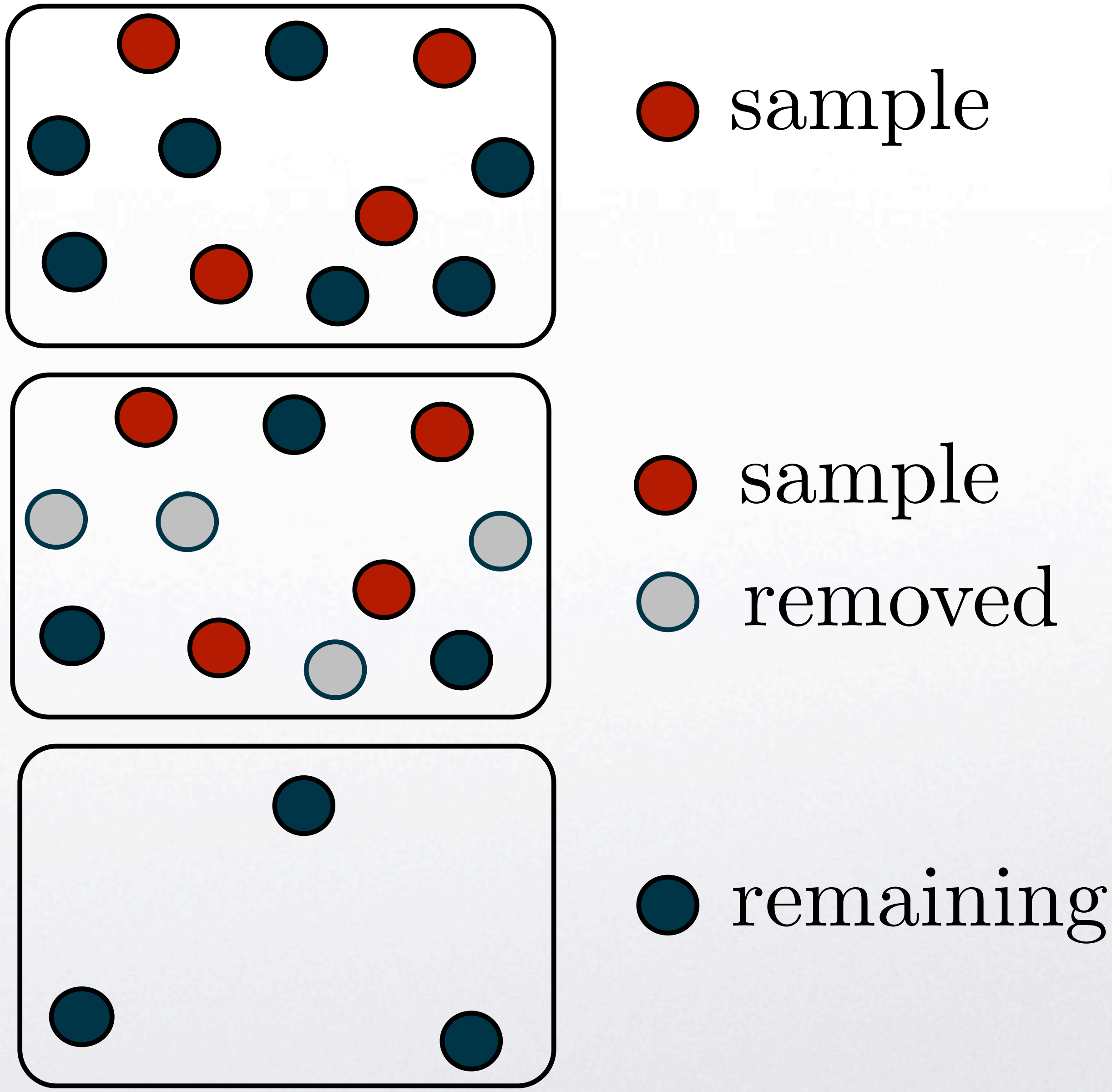
Memory/rounds trade-off?

Sampling Algorithm

- Construct a subset S of the points
- Points in S represent the input well
- Points in S fit on a single machine
- Use sampling to construct S

Sampling [also Thorup04]

- Sample $\tilde{\Theta}(n^\epsilon)$ points
- Add sample to S
- Remove an n^ϵ fraction of the points
  - Points removed are closest to the sample
- Apply procedure on remaining points



Sampling + kMedian

- Sample a subset S of the points
- Construct a weighted kMedian instance
- Put S and the weights on a single machine
- Run a sequential kMedian algorithm on S

Analysis

- Constant factor approximation
  - $10\alpha + 3$ approximation
- Number of rounds is $O(1/\epsilon)$
- Memory: $\tilde{\Theta}(N^{\epsilon})$
- Machines: $\Theta(N^{\frac{1}{2}-\epsilon})$
- In $\mathcal{MRC}^0$

Approximation Intuition

- Only need to show that the sampled points approximate the optimal solution
- Large clusters in the optimal solution have a point sampled from them ($\Omega(n^\delta)$ points)
- Small Clusters:
  - If a sampled point is close to the cluster then the contribution is small
  - If the whole cluster is far, then all of the cluster was never removed

Sampling vs Partitioning

- Partitioning
  - $\Theta(\sqrt{N})$ memory, $\Theta(N^{\frac{1}{4}})$ machines
  - Number of rounds is a small constant
  - Approximation is 3α for kMedian
- Sampling
  - $\Theta(N^{\epsilon})$ memory, $\Theta(N^{\frac{1}{2}-\epsilon})$ machines
  - Number of rounds is $O(1/\epsilon)$
  - Approximation is $10\alpha + 3$ for kMedian

Concluding Remarks

- Sparse input
  - Distances can be represented implicitly using $o(n^2)$ space
  - Ex: shortest path dist in a sparse graphs
- Experiments on real-world data

Thank You!
Questions?