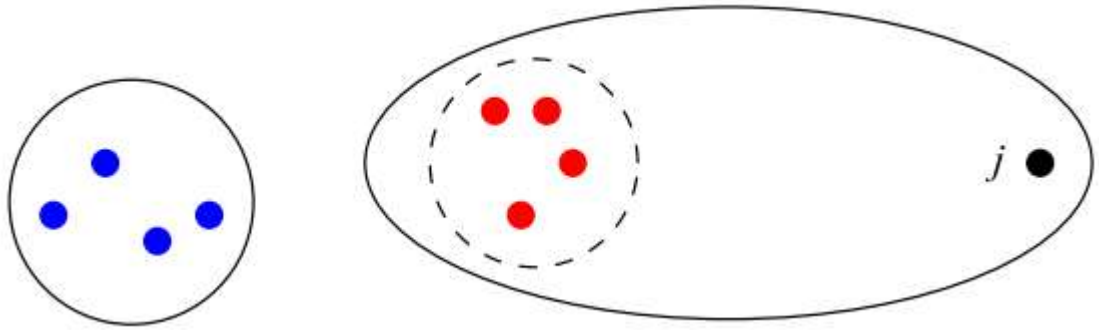
Fast Algorithms for Distributed k-Clustering with Outliers

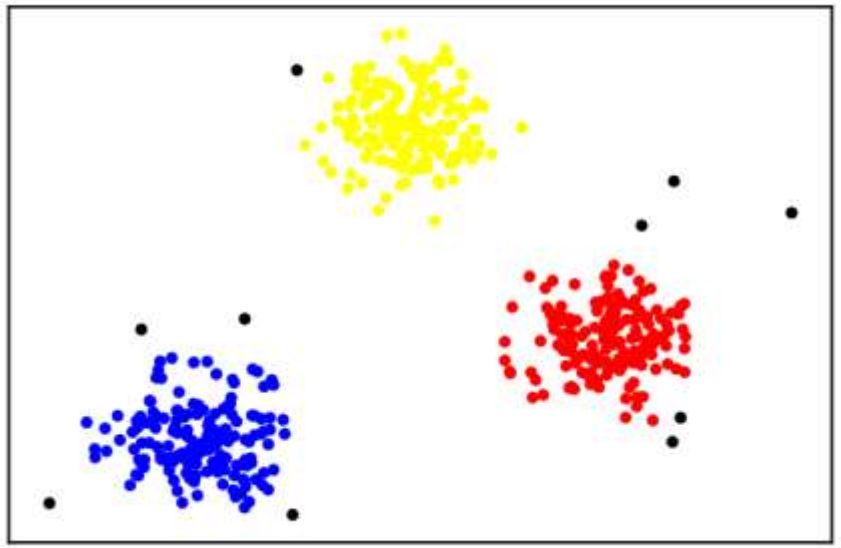
Junyu Huang Qilong Feng Ziyun Huang Jinhui Xu Jianxin Wang

Central South University

ICML 2023

Clustering with Outliers



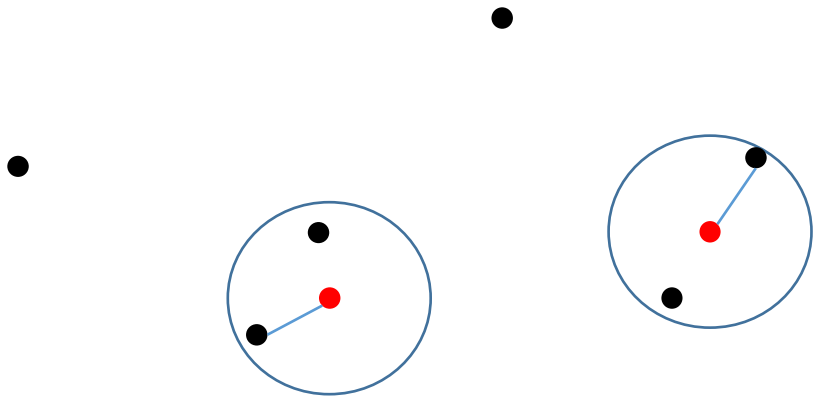


k-center with outlier

Input: a k-center instance (P, k), number of outliers z

Output: a set $C \subseteq P$ of at most k points with minimum clustering cost

$$(\max_{p \in P'} d(p, C), s.t. |P'| \geq n - z)$$

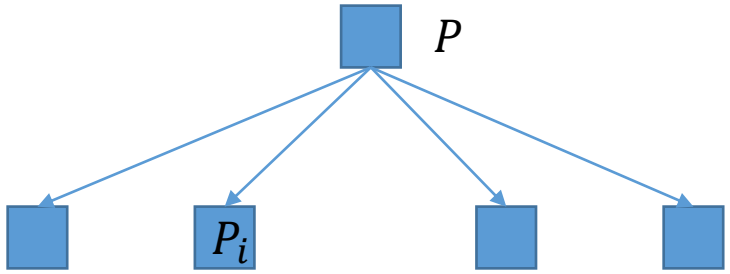


An instance for the k-center with outliers problem with $n = 8, z = 2, k = 2$

- Distributed clustering: in this setting, the dataset P is split among m machines, and P_i is the set of data points stored on machine i

- Communication cost: communications can only happen between the coordinator and the m machines. The communication is measured in the total number of words sent by machines and the coordinator

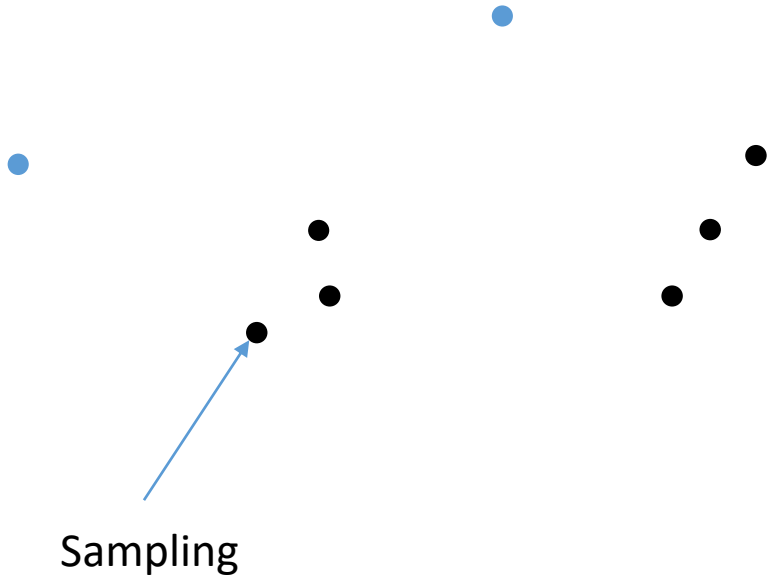
- Output: a set C of k centers, a bound L on maximum radius

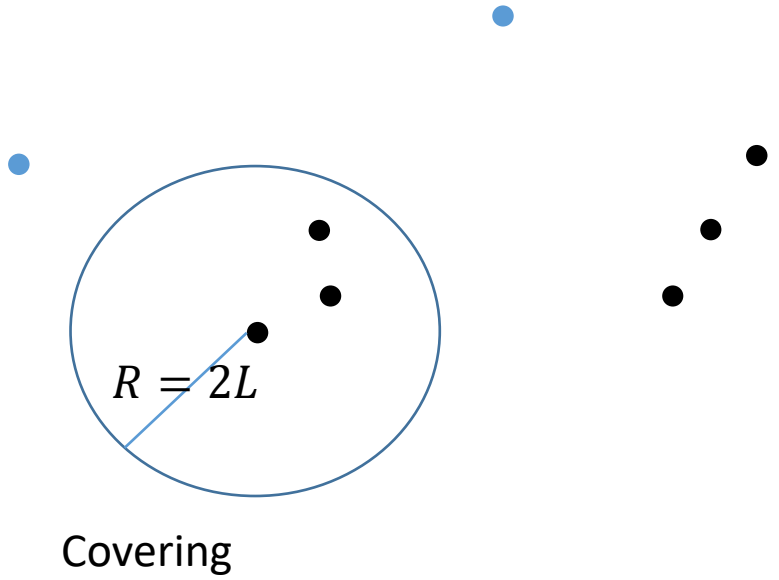


Distributed Algorithms

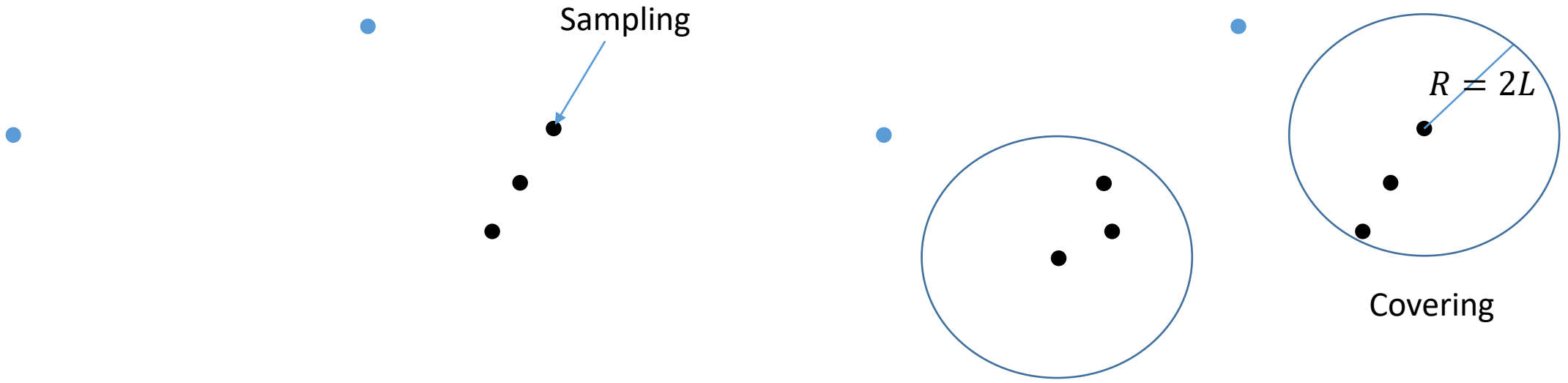
Method	Rounds	Approximation Ratio	Communication Cost	Local Time (Local computation time) (Coordinator time)	Reference
Greedy	2	$13(1+\epsilon)$	$O(m(k+z)log\Delta)$	$O(n_i klog\Delta)$ $O((k+z)^2 log\Delta)$	NeurIPS 2015
Convex Hull	2	$(O(1), 2+\epsilon)$	$O\left(\left(\frac{m}{\epsilon}+mk\right)log\Delta\right)$	$\tilde{O}(n_i^2)$ $\tilde{O}\left(\left(mk+\frac{m}{\epsilon}\right)^2\right)$	SPAA 2017
Aggregating	4	$(24(1+\epsilon), 1+\epsilon)$	$O\left(\frac{mk}{\epsilon}\frac{log\Delta}{\epsilon}\right)$	$\tilde{O}(n_i^2)$ $\tilde{O}\left(\left(\frac{mk}{\epsilon}\right)^2\right)$	NeurIPS 2018
Space-narrowing Sampling	2	$(14(1+\epsilon), 1+\epsilon)$	$O\left(\frac{mklogm}{\epsilon}\frac{log\Delta}{\epsilon}\right)$	$\left(\frac{n_i klogm}{\epsilon}\frac{log\Delta}{\epsilon}\right)$ $\tilde{O}\left(\left(\frac{mk}{\epsilon}\right)^2\right)$	**Ours**
Inliers-Recalling Sampling	4	$(14(1+\epsilon), 1+\epsilon)$	$O\left(\frac{m^3 klog(mk)}{\epsilon^2}\right)$	$O\left(\frac{n_i m^3 klog(mk)}{\epsilon^2}\right)$ $\tilde{O}\left(\left(\frac{m^3 k}{\epsilon^2}\right)^2\right)$	**Ours**

General Sampling Idea





General Sampling Idea



Our strategy

Single machine algorithm:

1. Check the remaining uncovered data size and gradually increase the sample size

2. Perform sampling and data removal

It can be guaranteed that

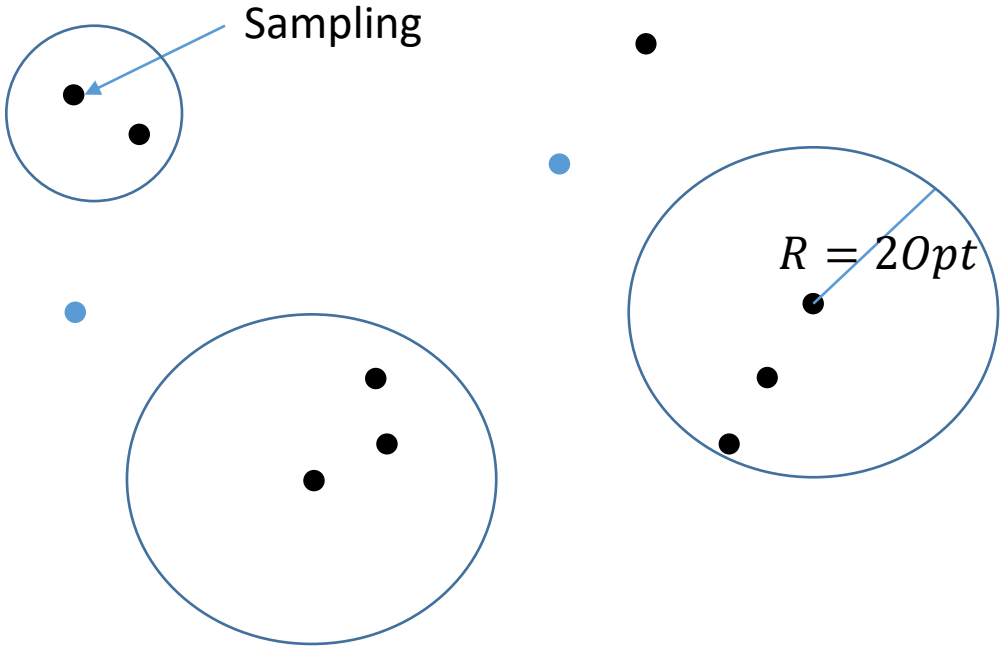
1. With constant probability, the number of uncovered inliers is at most ϵz

2. Each machine can find a 2-approximate solution with $O(k)$ centers opened

3. The communication cost can be bounded by $O(\frac{mk}{\epsilon}\frac{log\Delta}{\epsilon})$

Key Strategy: Avoid guessing an optimal radius Opt on each machine

Obtain a 2-approximation solution on each machine with $(1+\epsilon)z$ outliers discarded



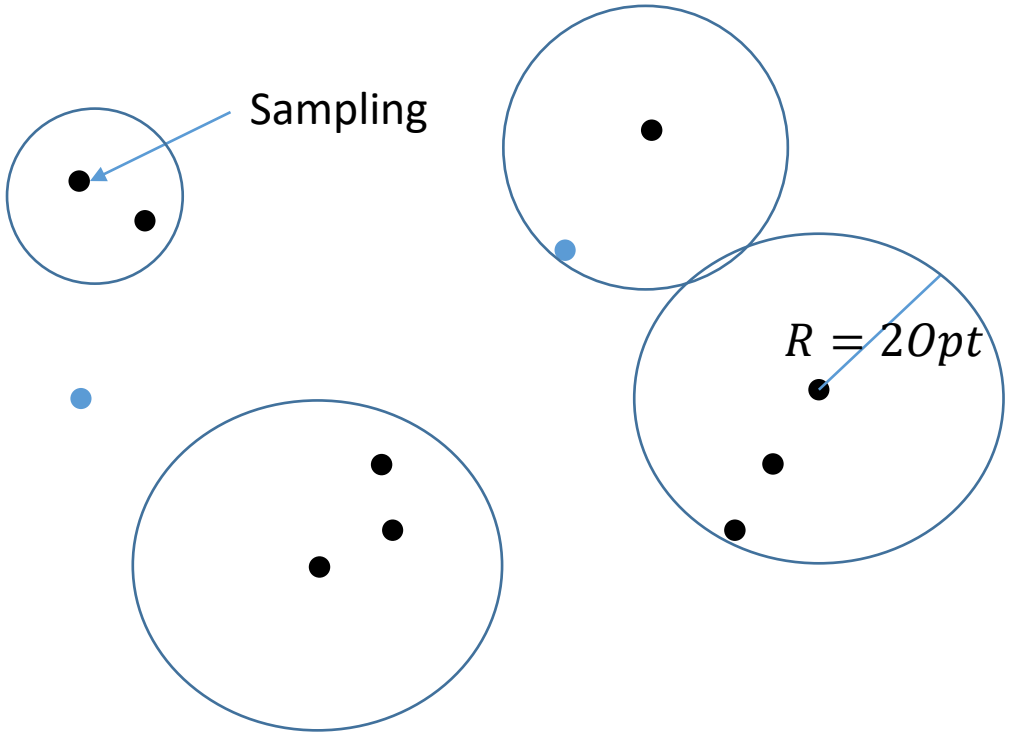
Uncovered Optimal Clusters:

1. $|C_i^*| \geq \frac{\epsilon z}{2km}$ (large uncovered clusters), which takes at least a fraction of $\frac{\epsilon}{(1+\epsilon)2km}$. In this case, sampling works for sample size $\frac{(1+\epsilon)2km}{\epsilon} log \frac{1}{\eta}$

2. $|C_i^*| < \frac{\epsilon z}{2km}$ (small uncovered clusters), can be discarded as outliers

Key Strategy: Avoid guessing an optimal radius Opt on each machine

Problem: We only have obtained the sampled centers, but we do not know the uncovered small clusters and outliers

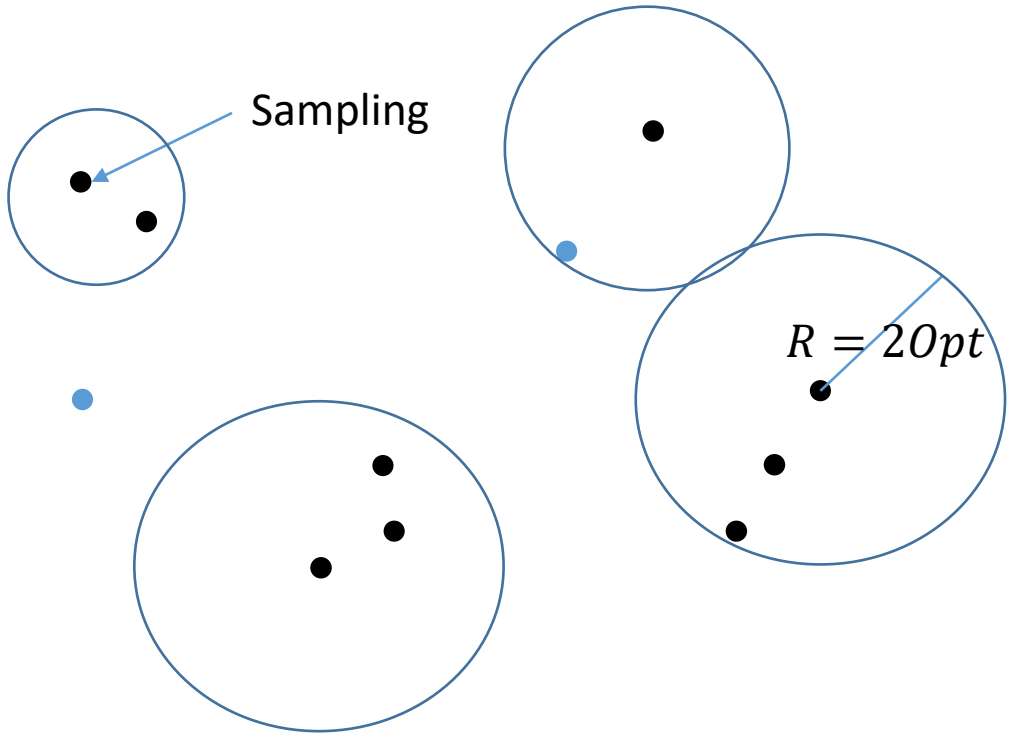


Strategy: Iteratively guessing the nearest $\frac{\epsilon z}{2m}$ data points in the discarded $(1+\epsilon)z$ data points as inliers.

We can find most inliers with distances smaller than $2Opt$ to the sampled centers.

Key Strategy: Avoid guessing an optimal radius Opt on each machine

Problem: We only have obtained the sampling centers, but we do not know the uncovered small clusters and outliers



We can now obtain $O\left(\frac{m}{\epsilon}\right)$ candidate radius for each machine and finally we only need to try all the radius.

Thanks