



# One-pass approximate *k*-means optimization

---

Claire Monteleoni  
Columbia University, CCLS

Joint work with Nir Ailon (Google)  
and Ragesh Jaiswal (Columbia)



# One-pass streaming setting

Streaming setting is similar to online setting, however data stream is finite.

Motivation: very large data-sets, and/or resource constraints (time, memory).

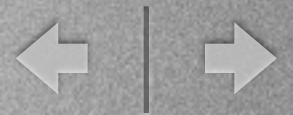
Goal: algorithms that are light-weight (time, memory), and make only one-pass over the data.

We study unsupervised learning, in the streaming setting.

Feedback is extremely limited: NO labels, but algorithm can compute intermediate values of the objective function, on points seen so far.







# *k*-means clustering objective

Clustering algorithms can be hard to evaluate without prior information or assumptions on the data.

With no assumptions on the data, one evaluation technique is w.r.t some **objective function**.

A widely-cited and studied objective is **k-means**: Given set,  $X \subset R^d$ , choose  $C \subset R^d$ ,  $|C| = k$ , to minimize: 
$$\phi_C = \sum_{x \in X} \min_{c \in C} \|x - c\|^2$$

Optimizing *k*-means is NP hard, even for  $k=2$  [DFKV '04].

Widely-used algorithm of same name [Lloyd '57]. Fast but lacks approximation guarantee, and can suffer from bad initialization.



# Related work

[Arthur & Vassilvitskii, SODA '07]: **k-means++**, a batch clustering algorithm with  $O(\log k)$ -approx. of  $k$ -means.

[Guha, Meyerson, Mishra, Motwani, & O'Callaghan, TKDE '03]: Divide and conquer streaming  $(a,b)$ -approximate  $k$ -medoid clustering.

Definition:  $b$ -approximation:  $\frac{\phi_C}{\phi_{OPT}} \leq b$

Definition: Bi-criteria  $(a,b)$ -approximation guarantee:  $a \cdot k$  centers,  $b$ -approx.





# Contributions

Extend  $k$ -means++ to  $k$ -means#, an  $(O(\log k), O(1))$ -approximation to  $k$ -means, in batch setting.

Analyze Guha *et al.* divide and conquer algorithm, using  $(a,b)$ -approximate  $k$ -means clustering.

Use Guha *et al.* with  $k$ -means# and then  $k$ -means++ to yield a **one-pass  $O(\log k)$ -approximation** algorithm to  $k$ -means objective.

Analyze multi-level hierarchy version for improved memory vs. approximation tradeoff.

Experiments on real and simulated data.



# $k$ -means++

## Algorithm:

Choose first center  $c_1$  uniformly at random from  $X$ ,  
and let  $C = \{c_1\}$ .

Repeat  $(k-1)$  times:

Choose next center  $c_i = x' \in X$  with prob.  $\frac{D(x', C)^2}{\sum_{x \in X} D(x, C)^2}$

$C \leftarrow C \cup \{c_i\}$

where  $D(x, C) = \min_{c \in C} \|x - c\|$

Theorem (Arthur & Vassilvitskii '07): Returns an  $O(\log k)$ -approximation, in expectation.





# $k$ -means#

Idea:  $k$ -means++ returns  $k$  centers, with  $O(\log k)$ -approximation. Can we design a variant that returns  $O(k \log k)$  centers, but **constant** approximation?

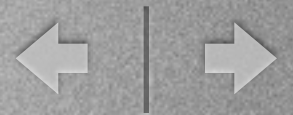
Algorithm:

Initialize  $C = \{\}$ .

Choose  $3 \cdot \log(k)$  centers independently and uniformly at random from  $X$ , and add them to  $C$ .

Repeat  $(k-1)$  times:

Choose  $3 \cdot \log(k)$  centers indep. with prob.  $\frac{D(x', C)^2}{\sum_{x \in X} D(x, C)^2}$  and add them to  $C$ .



# $k$ -means#

**Theorem:** With probability at least  $1/4$ ,  $k$ -means# yields an  $O(1)$ -approximation, on  $O(k \log k)$  centers.

**Corollary:** With probability at least  $1 - 1/n$ , running  $k$ -means# for  $3 \cdot \log n$  independent runs yields an  $O(1)$ -approximation (on  $O(k \log k)$  centers).

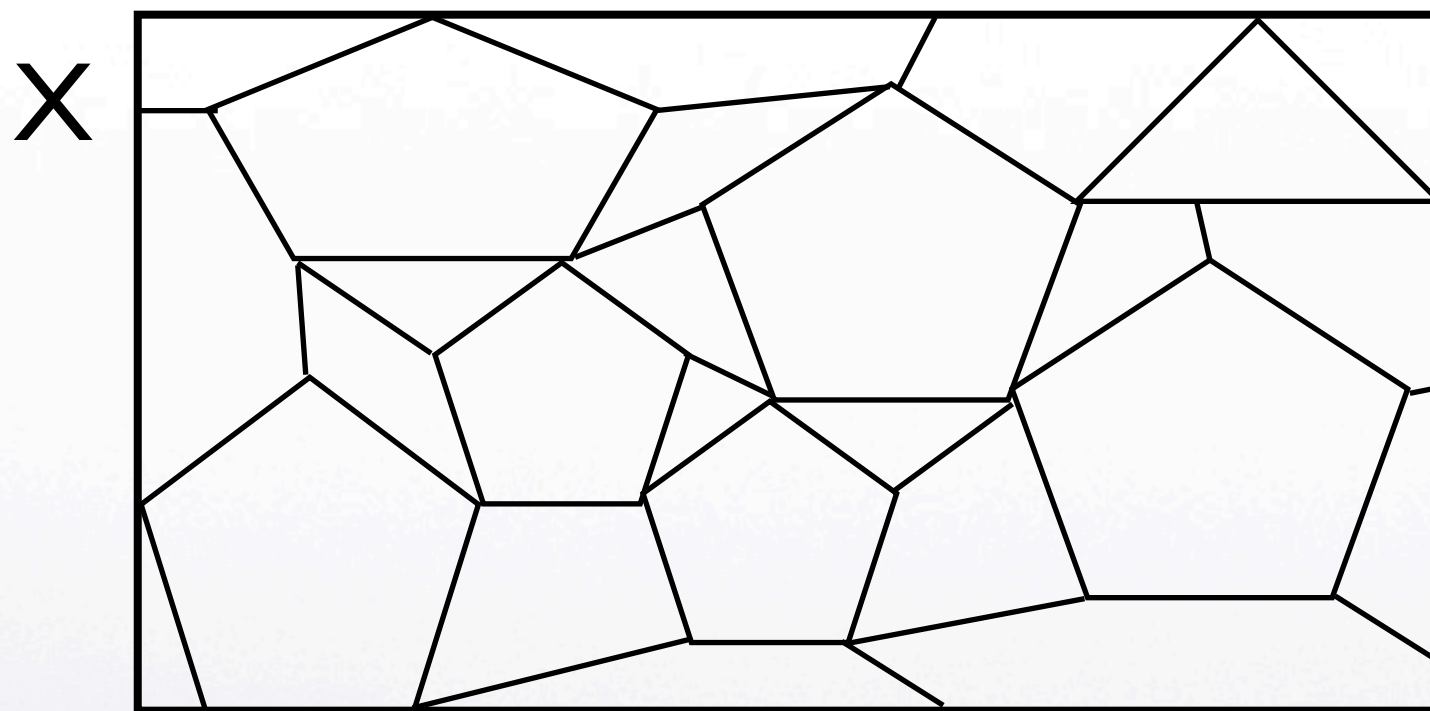
**Proof:** Call it repeatedly,  $3 \cdot \log n$  times, independently, and choose the clustering that yields the minimum cost. Corollary follows, since

$$(1 - (3/4)^{3 \log n}) \geq \left(1 - \frac{1}{n}\right).$$





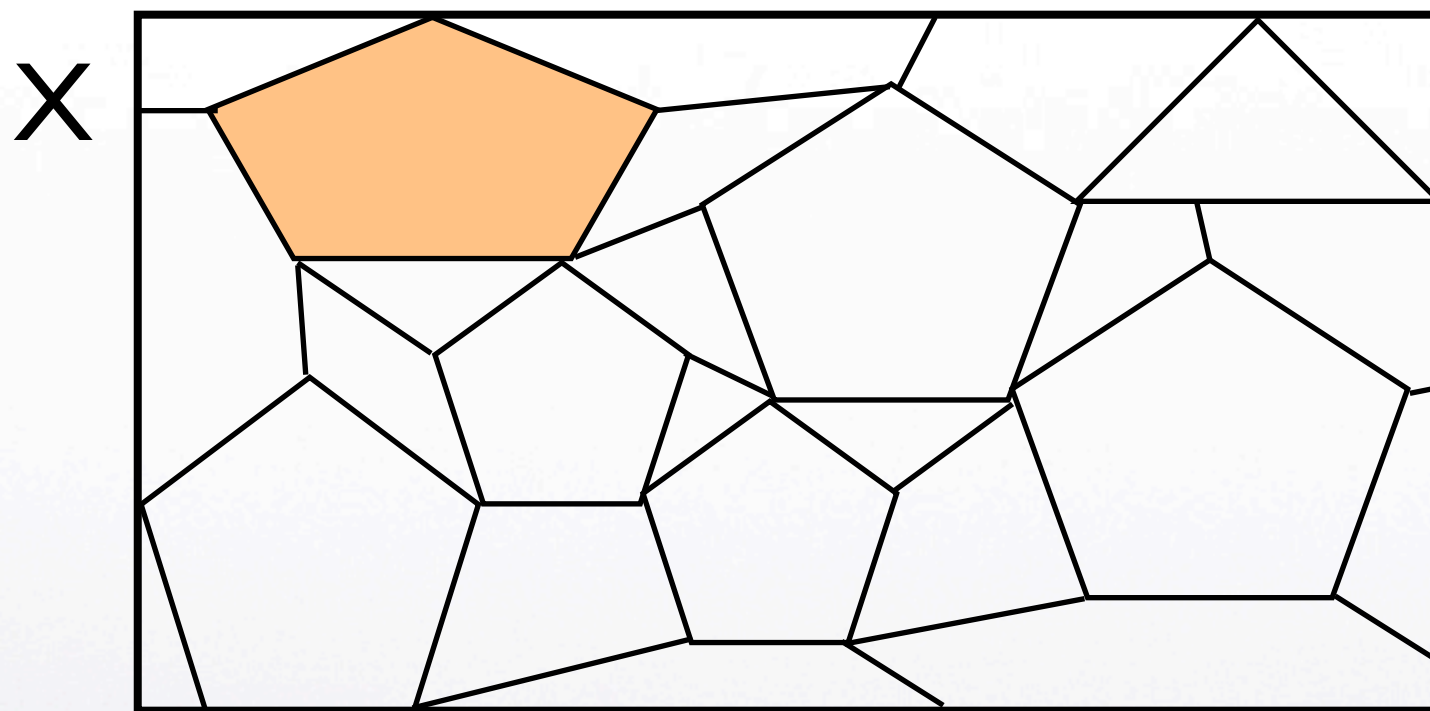
# $k$ -means# proof idea



The clustering (partition) induced by OPT.



# $k$ -means# proof idea

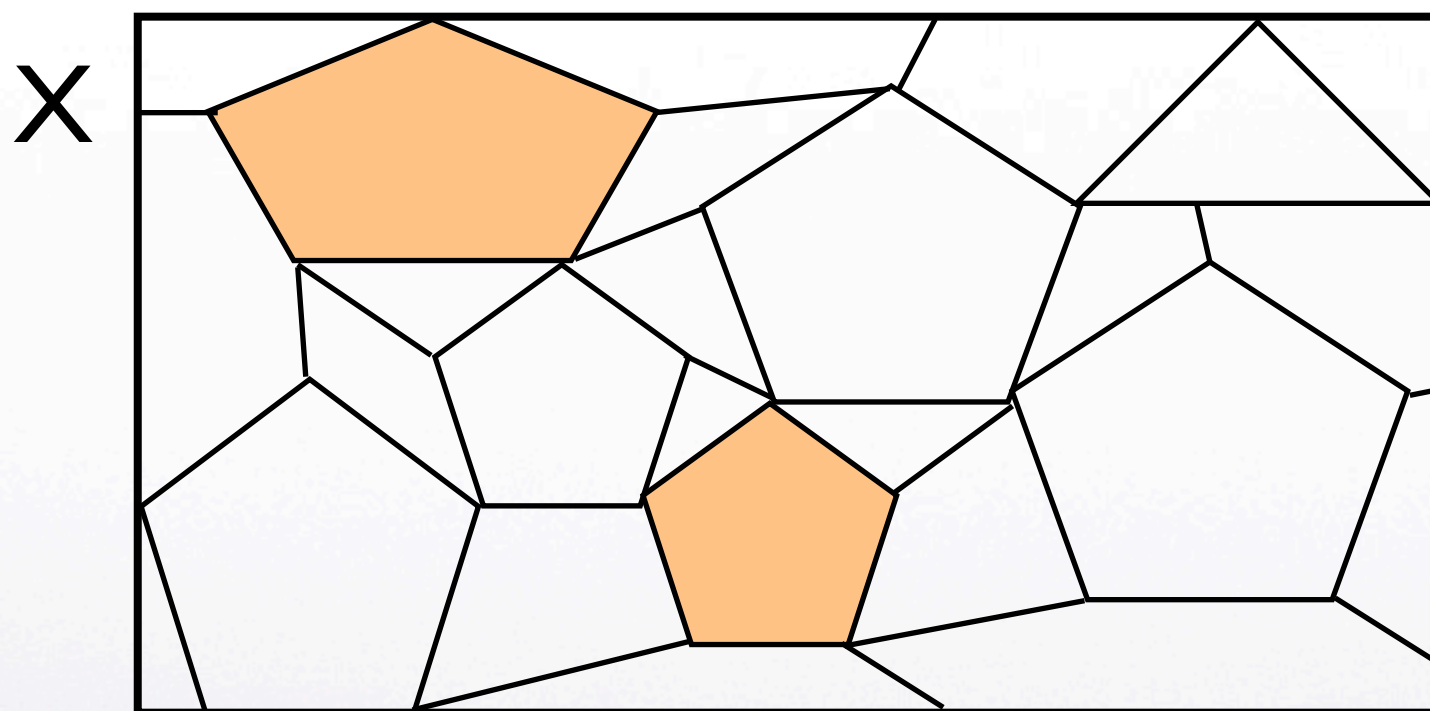


The clustering (partition) induced by OPT.





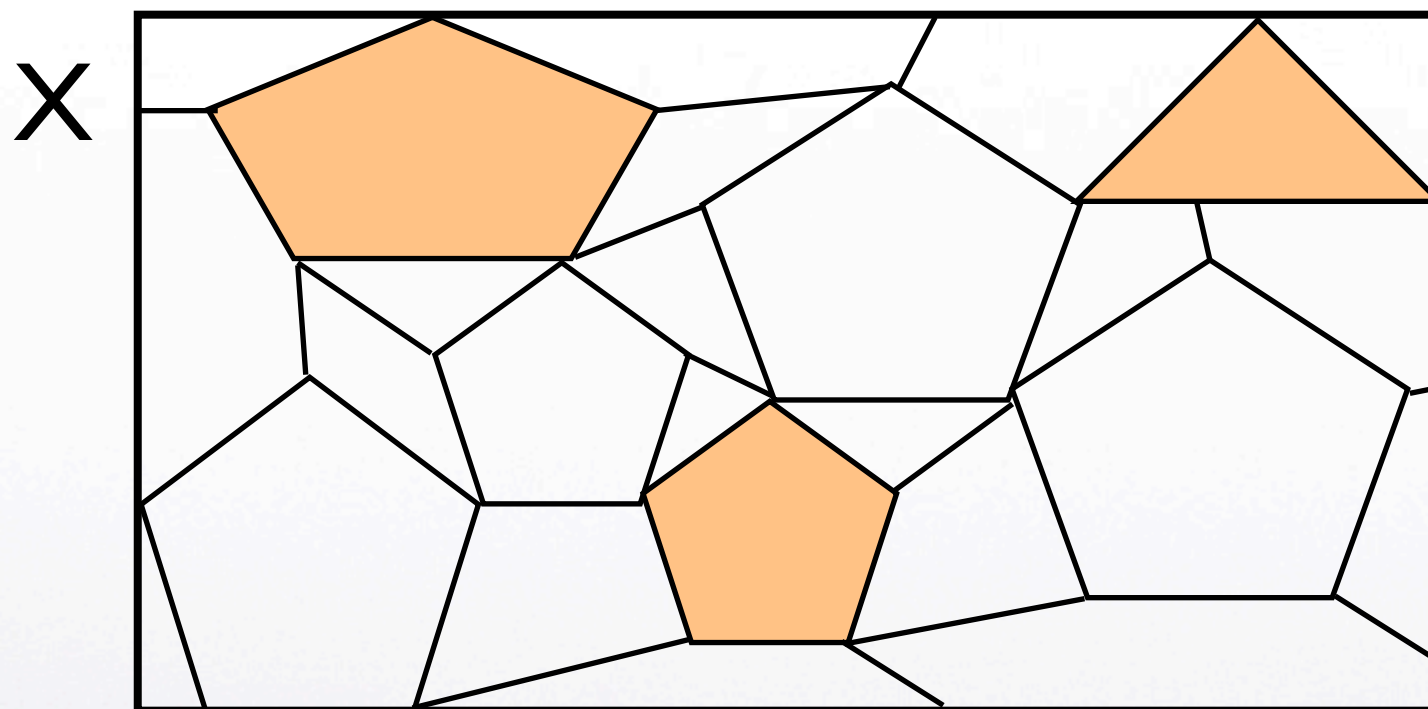
# $k$ -means# proof idea



The clustering (partition) induced by OPT.



# $k$ -means# proof idea

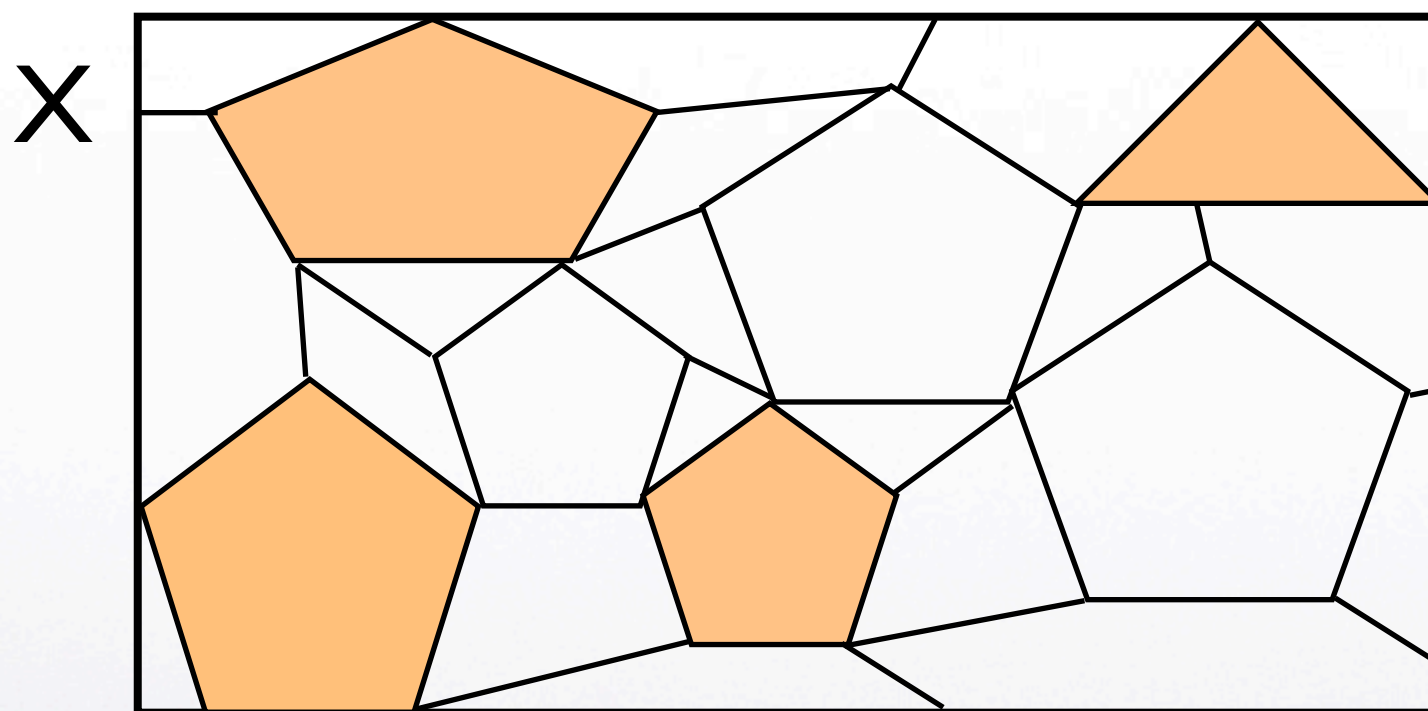


The clustering (partition) induced by OPT.





# $k$ -means# proof idea



The clustering (partition) induced by OPT.

→ We cover the  $k$  clusters in OPT, after choosing  $O(k \log k)$  centers.



# $k$ -means#

**Theorem:** With probability at least  $1/4$ ,  $k$ -means# yields an  $O(1)$ -approximation, on  $O(k \log k)$  centers.

**Proof outline:** Definition “covered”: cluster  $A \in \text{OPT}$  is covered if:  $\phi_C(A) < 32 \cdot \phi_{\text{OPT}}(A)$ , where  $\phi_C(A) = \sum_{x \in A} D(x, C)^2$ .

Define  $\{X_c, X_u\}$ : the partition of  $X$  into covered, uncovered.

In the first round we cover one cluster in  $\text{OPT}$ . In any later round, either:

**Case 1:**  $\phi_C(X_c) > \phi_C(X_u)$ : We are done.

**Case 2 :**  $\phi_C(X_c) \leq \phi_C(X_u)$ : We are likely to cover another  $\text{OPT}$  cluster.





# $k$ -means# proof

Fix any point  $x$  chosen in the first step. Define  $A$  as the unique cluster in  $OPT$ , s.t.  $x \in A$ .

Lemma (AV '07): Fix  $A \in OPT$ , and let  $C$  be the 1-clustering with the center chosen uniformly at random from  $A$ . Then  $E[\phi_C(A)] = 2 \cdot \phi_{OPT}(A)$ .

Corollary:  $Pr[\phi_C(A) < 8 \cdot \phi_{OPT}(A)] \geq 3/4$ . Pf. Apply Markov's inequality.

After  $3 \cdot \log(k)$  random points, probability of hitting a cluster  $A$  with a point that is good for  $A$  is at least  $(1-1/k)$ .

So after first step, w.p. at least  $(1-1/k)$ , at least 1 cluster is covered.



# *k*-means# proof

Case 1:  $\phi_C(X_c) > \phi_C(X_u)$ .

Since  $X = X_c \cup X_u$  and by definition of  $\phi$ ,

$$\phi_C(X) = \phi_C(X_c) + \phi_C(X_u) \leq 2 \cdot \phi_C(X_c) \leq 64 \cdot \phi_{OPT}(X_c) \leq 64 \cdot \phi_{OPT}(X)$$

by definition of Case 1, and definition of covered.

Last inequality is by  $X_c \subseteq X$ , and definition of  $\phi$ .





# $k$ -means# proof

Case 2:  $\phi_C(X_c) \leq \phi_C(X_u)$ .

The probability of picking a point in  $X_u$  at the next round is:

$$\frac{\sum_{x \in X_u} D(x, C)^2}{\sum_{x \in X} D(x, C)^2} = \frac{\phi_C(X_u)}{\phi_C(X_u) + \phi_C(X_c)} \geq \frac{1}{2}$$

**Lemma (AV '07):** Fix  $A \in \text{OPT}$ , and let  $C$  be any clustering. If we add a center to  $C$ , sampled randomly from the  $D^2$  weighting over  $A$ , yielding  $C'$  then:  $E[\phi_{C'}(A)] \leq 8 \cdot \phi_{\text{OPT}}(A)$ . **Corollary:**  $\Pr[\phi_{C'}(A) < 32 \cdot \phi_{\text{OPT}}(A)] \geq 3/4$

So, w.p.  $\geq \frac{1}{2} \cdot \frac{3}{4} = \frac{3}{8}$  we pick a point in  $X_u$  that covers a new cluster in  $\text{OPT}$

So after  $3 \cdot \log(k)$  picks, prob. of covering a new cluster is at least  $(1 - 1/k)$ .



# $k$ -means# proof summary

For the first round, prob. of covering a cluster in OPT is at least  $(1-1/k)$ .

For the  $k-1$  remaining rounds, either Case 1 holds, and we have achieved a 64-approximation, or Case 2 holds, and the probability of covering a new cluster in OPT, in the next round, is at least  $(1-1/k)$ .

So the probability that after  $k$  rounds there exists an uncovered cluster in OPT is  $\leq 1 - (1 - 1/k)^k \leq 3/4$ .

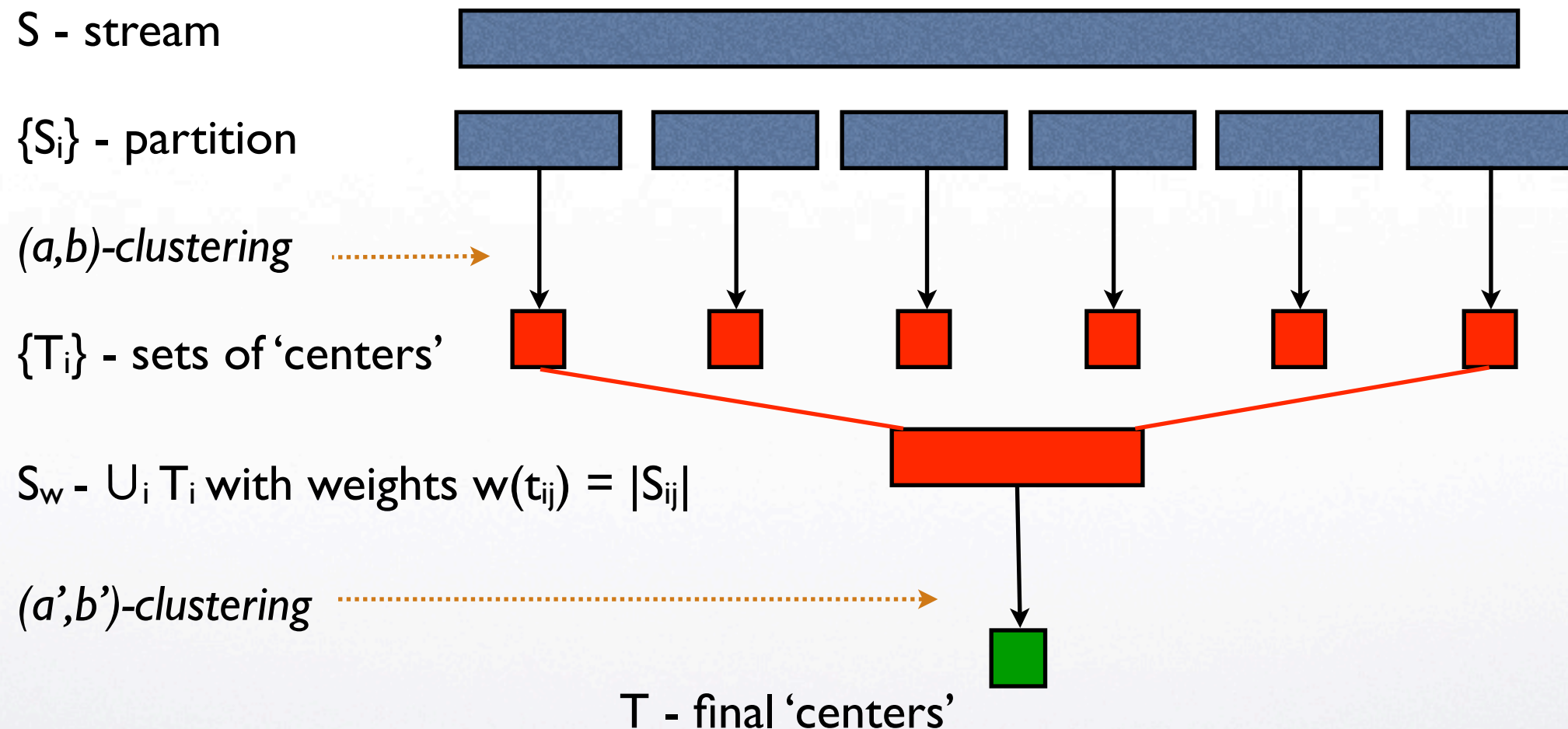
Thus the algorithm achieves a 64-approximation on  $3k \cdot \log(k)$  centers, with probability at least  $1/4$ .

Corollary: Repeating it  $3 \cdot \log(n)$  times yields probability  $(1-1/n)$ .

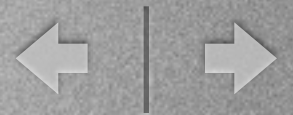




# Divide and conquer clustering



[Guha et al. '03] analyzed for  $k$ -medoid clustering:  $(a', O(bb'))$ -approximation.



# One-pass $k$ -means approx.

We first analyze Guha *et al.* scheme for  $(a,b)$ -approximation algorithms w.r.t.  $k$ -means: yields a one-pass  $(a', O(bb'))$ -approximation algorithm.

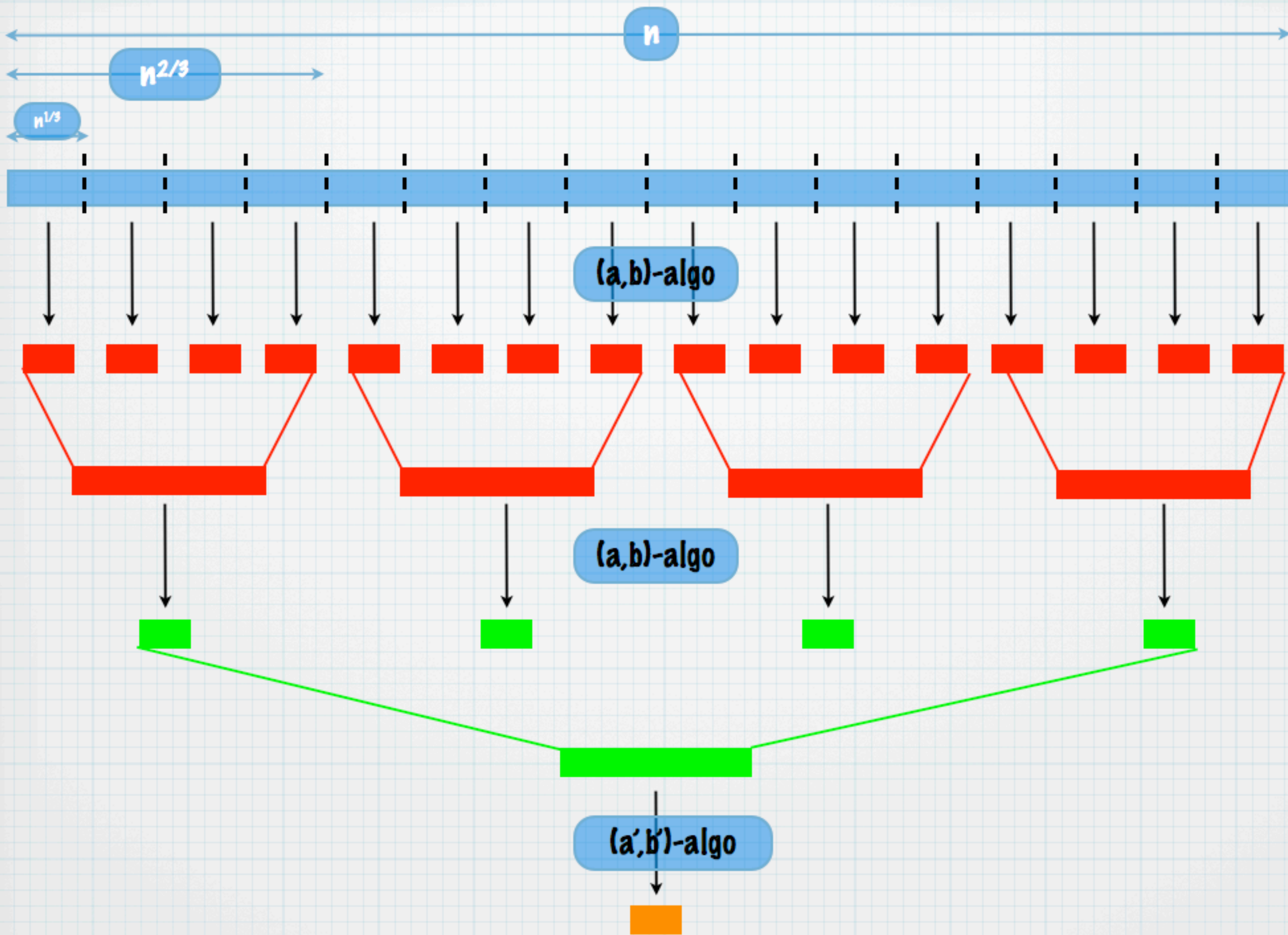
Our algorithm:

For the  $(a,b)$  algorithm, use (repeated)  $k$ -means#:  $a = O(\log k)$ ,  $b = O(1)$ .

For the  $(a',b')$  algorithm, use  $k$ -means++:  $a' = 1$ ,  $b' = O(\log k)$

So the combined algorithm is a  $(1, O(\log k))$ -approximation to  $k$ -means.







# Memory vs. approximation

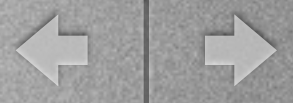
Generalize to multi-level hierarchy; idea in Guha *et al.*

Call repeated  $k$ -means# at all levels but the last, and  $k$ -means++ at the last.

Theorem: Given memory  $M = n^\alpha$  for a fixed  $\alpha > 0$ , letting  $r = 1/\alpha$  yields an  $r$ -level one-pass algorithm with  $O(c^{r-1} \log k)$ -approximation.

Note: Unit of memory is a word; a point in  $R^d$  can be stored in  $O(1)$  space.





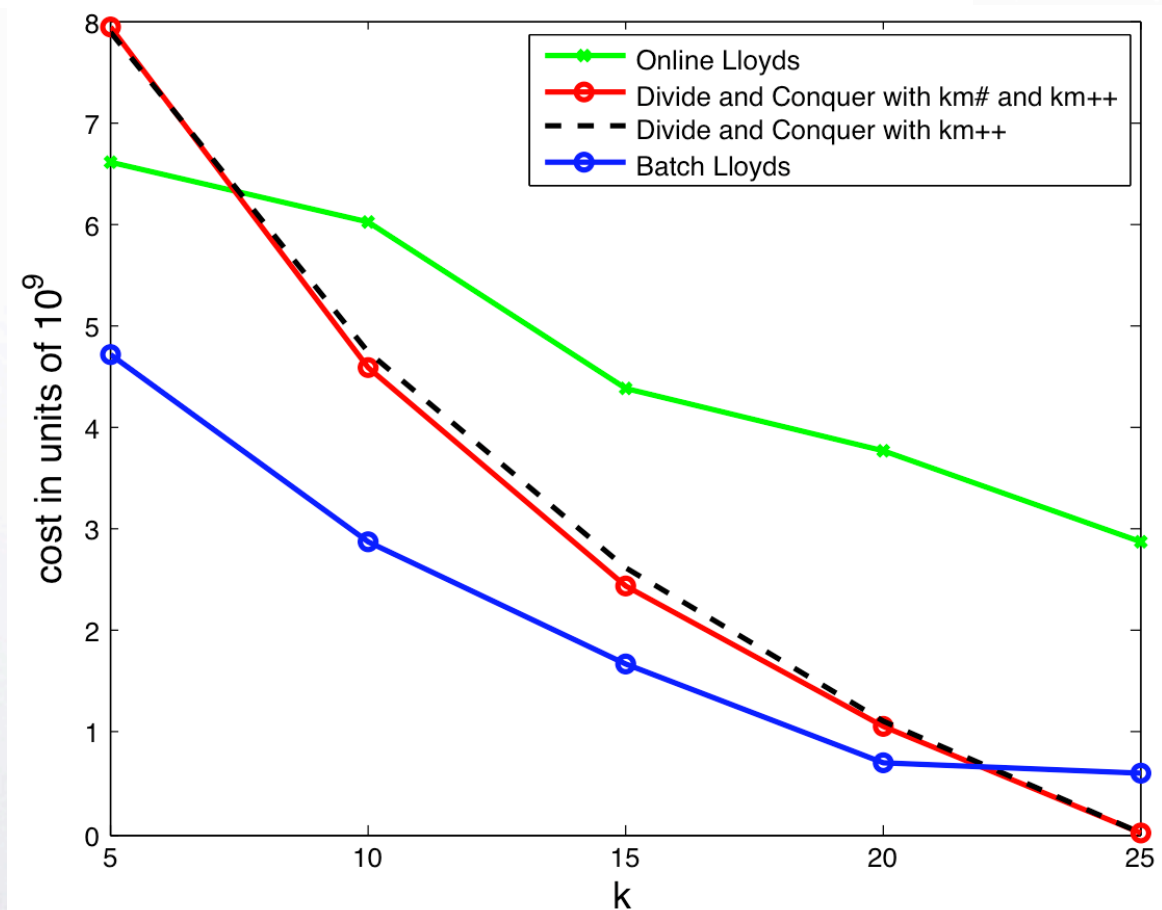
# Experiments

k	BL	OL	DC-1	DC-2	BL	OL	DC-1	DC-2
5	$4.7254 \cdot 10^9$	$6.5967 \cdot 10^9$	$7.9336 \cdot 10^9$	$7.8752 \cdot 10^9$	5.80	1.44	16.95	12.22
10	$2.8738 \cdot 10^9$	$6.0146 \cdot 10^9$	$4.5968 \cdot 10^9$	$4.7288 \cdot 10^9$	7.33	2.76	53.10	24.74
15	$1.6753 \cdot 10^9$	$4.3743 \cdot 10^9$	$2.4338 \cdot 10^9$	$2.6280 \cdot 10^9$	8.85	4.00	112.68	36.86
20	$7.0016 \cdot 10^8$	$3.7794 \cdot 10^9$	$1.0661 \cdot 10^9$	$1.1017 \cdot 10^9$	11.75	6.04	250.21	48.57
25	$6.0011 \cdot 10^8$	$2.8859 \cdot 10^9$	$2.7493 \cdot 10^5$	$2.7906 \cdot 10^5$	13.83	7.00	403.81	60.96

Table 1: norm25 dataset. (columns 2-5 has the clustering cost and columns 6-9 has time in sec.)

## Mixture of 25 Gaussians:

10K points sampled from a mixture of 25 Gaussians chosen at random from 15 dimensional hypercube (side 500).

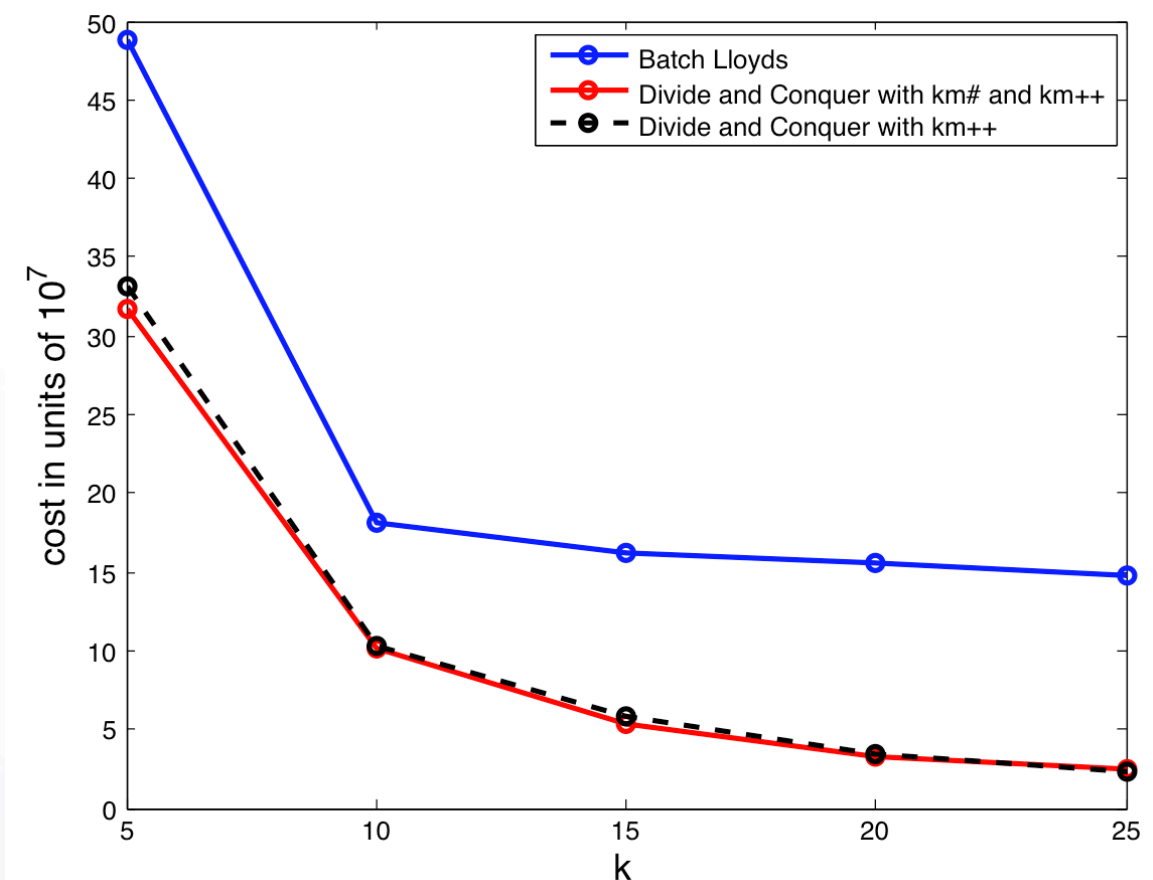
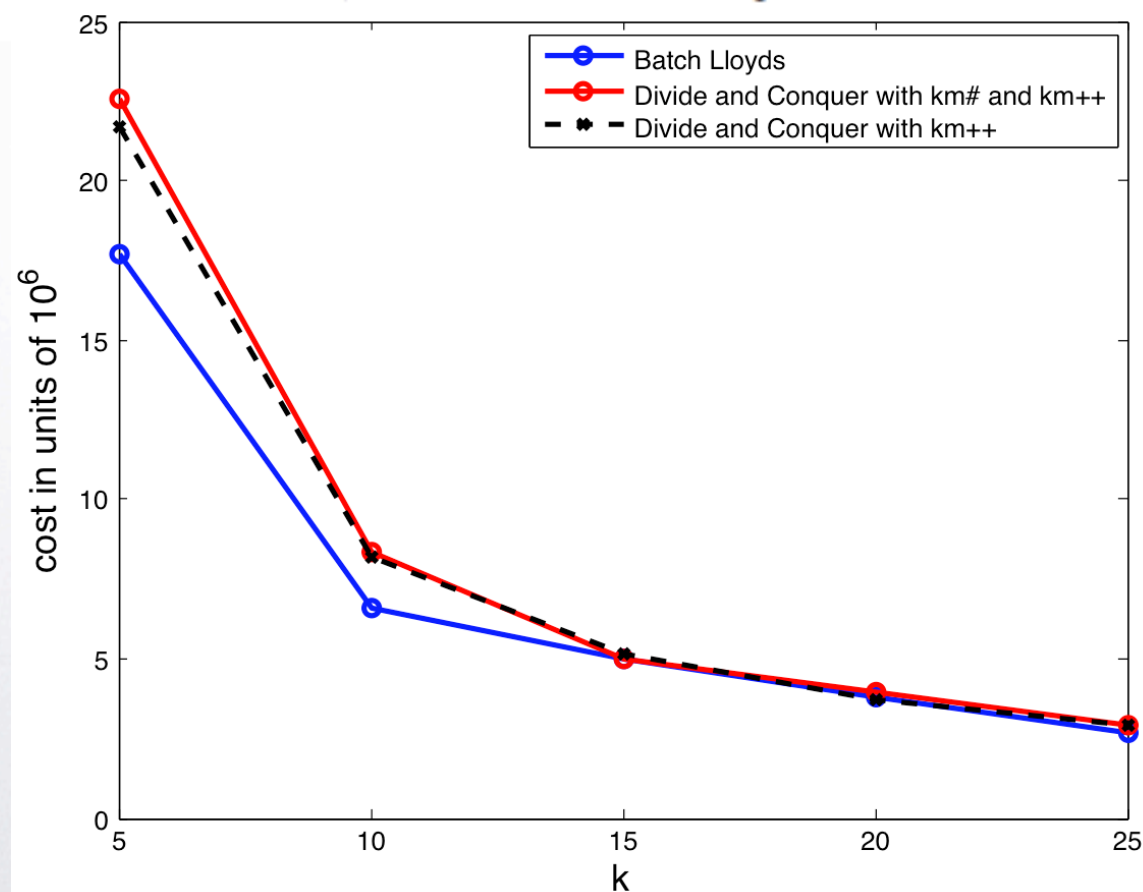




# Experiments

k	BL	OL	DC-1	DC-2	BL	OL	DC-1	DC-2
5	$1.7713 \cdot 10^7$	$1.2401 \cdot 10^8$	$2.2582 \cdot 10^7$	$2.1683 \cdot 10^7$	1.78	0.15	2.30	1.10
10	$6.5871 \cdot 10^6$	$8.5684 \cdot 10^7$	$8.3452 \cdot 10^6$	$8.2037 \cdot 10^6$	2.27	0.31	7.45	2.40
15	$4.9851 \cdot 10^6$	$8.4633 \cdot 10^7$	$4.9935 \cdot 10^6$	$5.1391 \cdot 10^6$	3.42	0.45	13.34	3.32
20	$3.7836 \cdot 10^6$	$6.5110 \cdot 10^7$	$3.9289 \cdot 10^6$	$3.7279 \cdot 10^6$	3.38	0.59	32.42	5.00
25	$2.6363 \cdot 10^6$	$6.3758 \cdot 10^7$	$2.8899 \cdot 10^6$	$2.9470 \cdot 10^6$	4.54	0.62	46.45	5.89

Table 2: Cloud dataset. (columns 2-5 has the clustering cost and columns 6-9 has time in sec.)



k	BL	OL	DC-1	DC-2	BL	OL	DC-1	DC-2
5	$4.8769 \cdot 10^8$	$1.7001 \cdot 10^9$	$3.1770 \cdot 10^8$	$3.3191 \cdot 10^8$	3.74	0.87	14.60	6.53
10	$1.8169 \cdot 10^8$	$1.6930 \cdot 10^9$	$1.0104 \cdot 10^8$	$1.0271 \cdot 10^8$	5.59	1.66	47.92	12.17
15	$1.6227 \cdot 10^8$	$1.4762 \cdot 10^9$	$5.3517 \cdot 10^7$	$5.7865 \cdot 10^7$	7.04	2.19	86.54	17.53
20	$1.5580 \cdot 10^8$	$1.4766 \cdot 10^9$	$3.2577 \cdot 10^7$	$3.4155 \cdot 10^7$	9.87	2.83	218.95	25.70
25	$1.4704 \cdot 10^8$	$1.4754 \cdot 10^9$	$2.3981 \cdot 10^8$	$2.2735 \cdot 10^8$	13.26	4.41	331.77	40.64

Table 3: Spambase dataset. (columns 2-5 has the clustering cost and columns 6-9 has time in sec.)

UCI data: Clouds and Spambase.





# Experiments

## Memory/approximation tradeoff:

Memory/ #levels	Cost	Time	Memory/ #levels	Cost	Time	Memory/ #levels	Cost	Time
1024/0	$8.74 \cdot 10^6$	5.5	2048/0	$5.78 \cdot 10^4$	30	4601/0	$1.06 \cdot 10^8$	34
480/1	$8.59 \cdot 10^6$	3.6	1250/1	$5.36 \cdot 10^4$	25	880/1	$0.99 \cdot 10^8$	20
360/2	$8.61 \cdot 10^6$	3.8	1125/2	$5.15 \cdot 10^4$	26	600/2	$1.03 \cdot 10^8$	19.5

UCI Cloud,  $k=10$ ; 25 Gaussians,  $k=25$ ; UCI Spambase,  $k=10$

Divide and conquer with  $k$ -means# at all levels except last, and then  $k$ -means++

These runs did not seem to reach a memory limit that would result in worse approximation.



# Future work

Simple extensions: tightening analysis, further experimentation.

Use [Kanungo *et al.* '04] as sub-algorithm to attain a one-pass algorithm with *constant* approximation.

Analyze under data assumptions, e.g. i.i.d. or well-separated means.

Next step: an algorithm that approximates  $k$ -means in the **online** setting.





# Thank you!

*And many thanks to my coauthors:*

Nir Ailon, Google Research NYC

Ragesh Jaiswal, Columbia University