Scalable K-Means++

Bahman Bahmani

Stanford University

K-means Clustering

- Fundamental problem in data analysis and machine learning
- "By far **the most popular clustering algorithm** used in scientific and industrial applications" [Berkhin '02]
- Identified as one of the **top 10 algorithms in data mining** [Wu et al '07]

Problem Statement

- A scalable algorithm for K-means clustering with theoretical guarantees and good practical performance

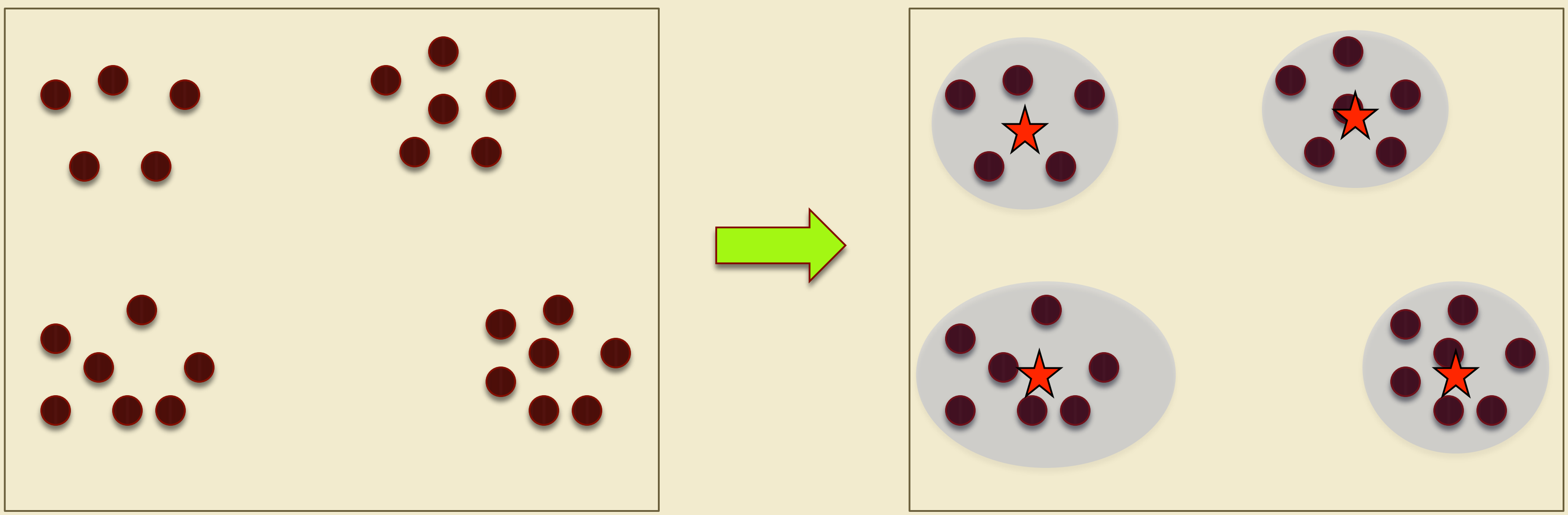
K-means Clustering

- **Input:**
  - A set $X=\{x_1, x_2, \ldots, x_n\}$ of n data points
  - Number of clusters k
- For a set $C=\{c_1, c_2, \ldots, c_k\}$ of cluster "centers" define:

$$\varphi_X(C) = \sum_{x \in X} d(x,C)^2$$

  where $d(x,C)$ = distance from x to closest center in C
- **Goal:** To find a set C of centers that minimizes the objective function $\varphi_X(C)$

K-means Clustering: Example



K = 4

Lloyd Algorithm

- Start with k arbitrary centers $\{c_1, c_2, \ldots, c_k\}$ (typically chosen uniformly at random from data points)
- Performs an EM-type local search till convergence
- **Main advantages:** Simplicity, scalability

What's wrong with Lloyd Algorithm?

- Takes many iterations to converge
- Very sensitive to initialization
- Random initialization can easily get two centers in the same cluster
  - K-means gets stuck in a local optimum

Lloyd Algorithm: Initialization

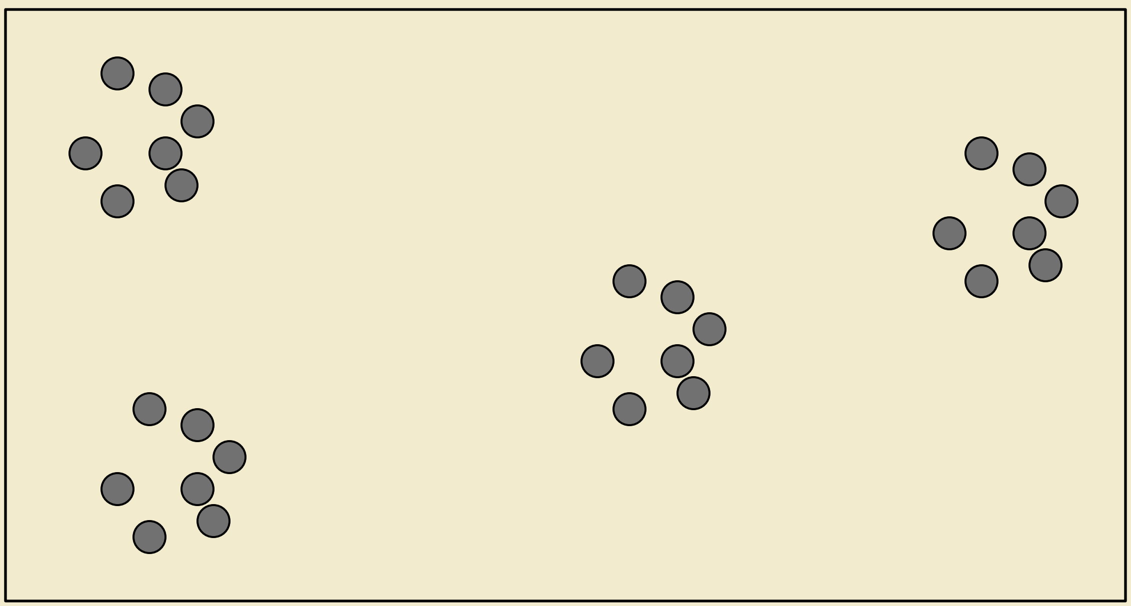


Figure credited to David Arthur

Lloyd Algorithm: Initialization

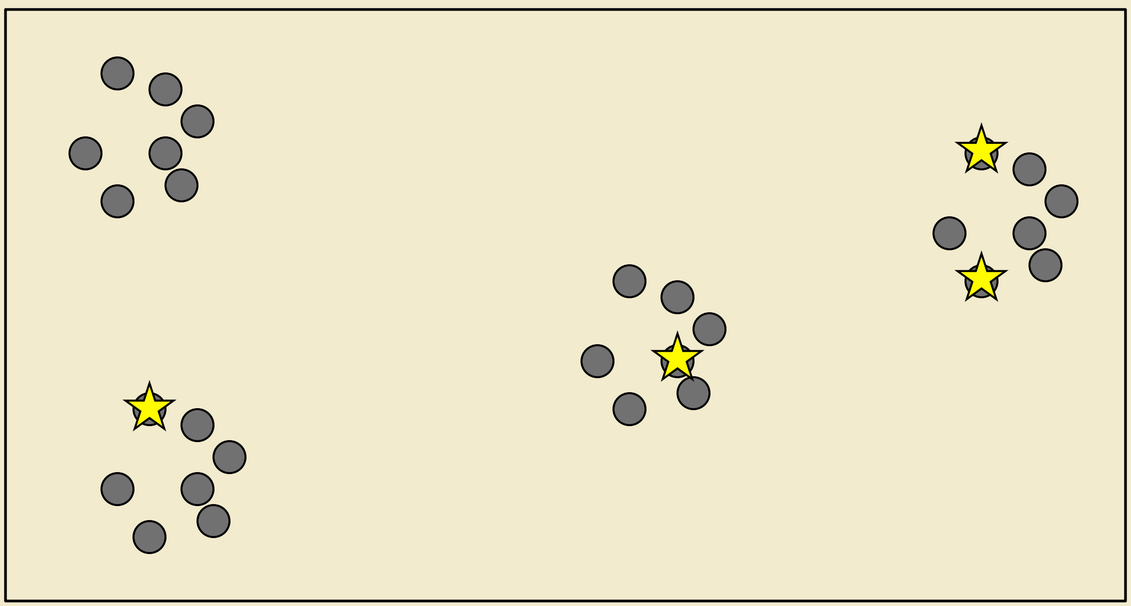


Figure credited to David Arthur

Lloyd Algorithm: Initialization

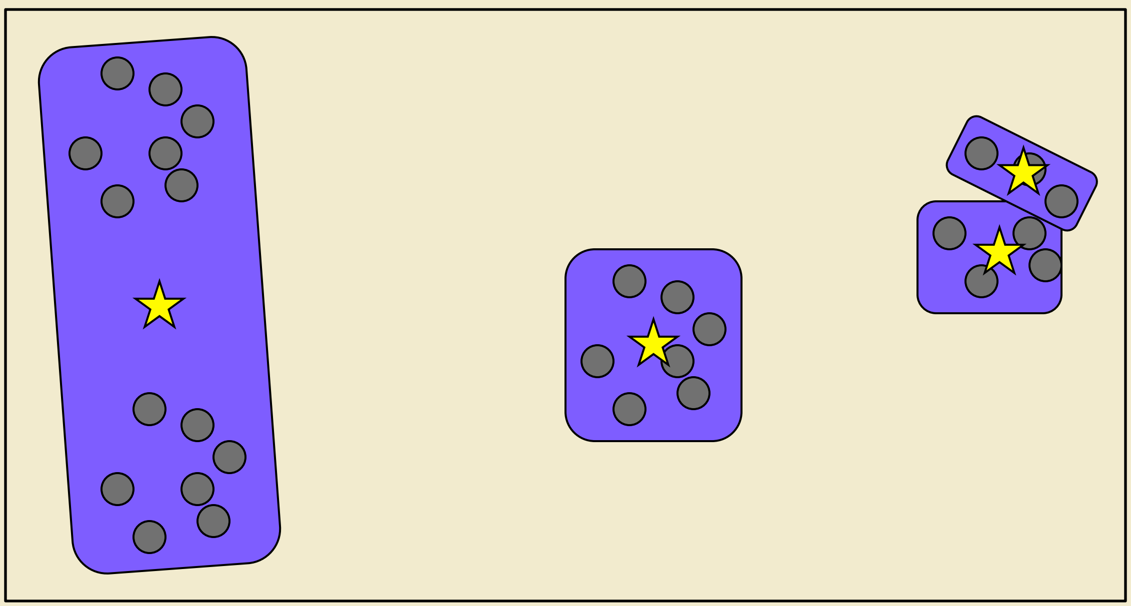


Figure credited to David Arthur

Lloyd Algorithm: Initialization

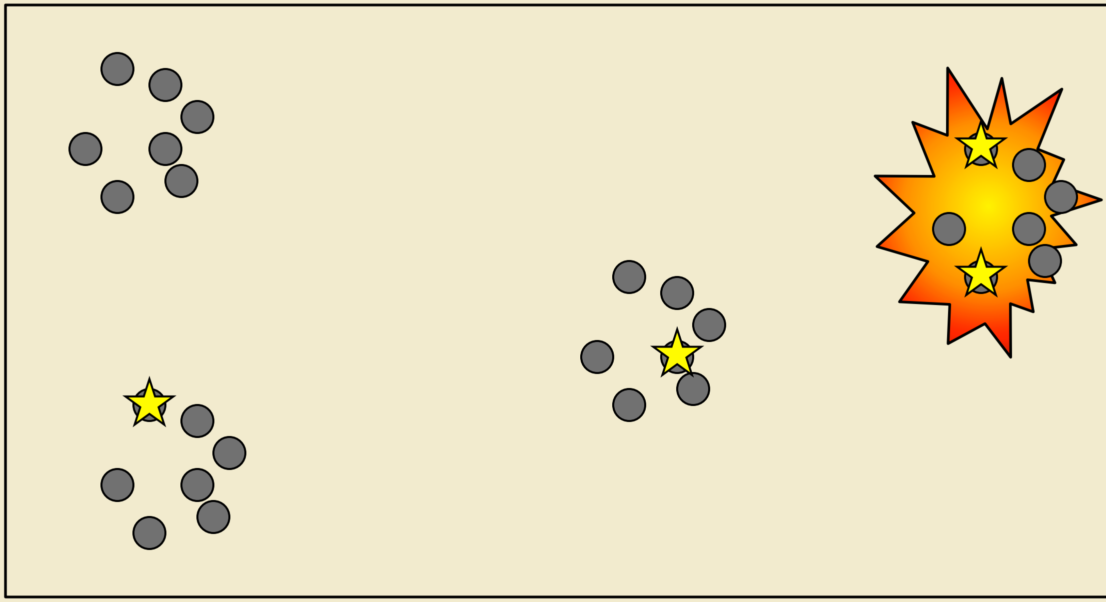


Figure credited to David Arthur

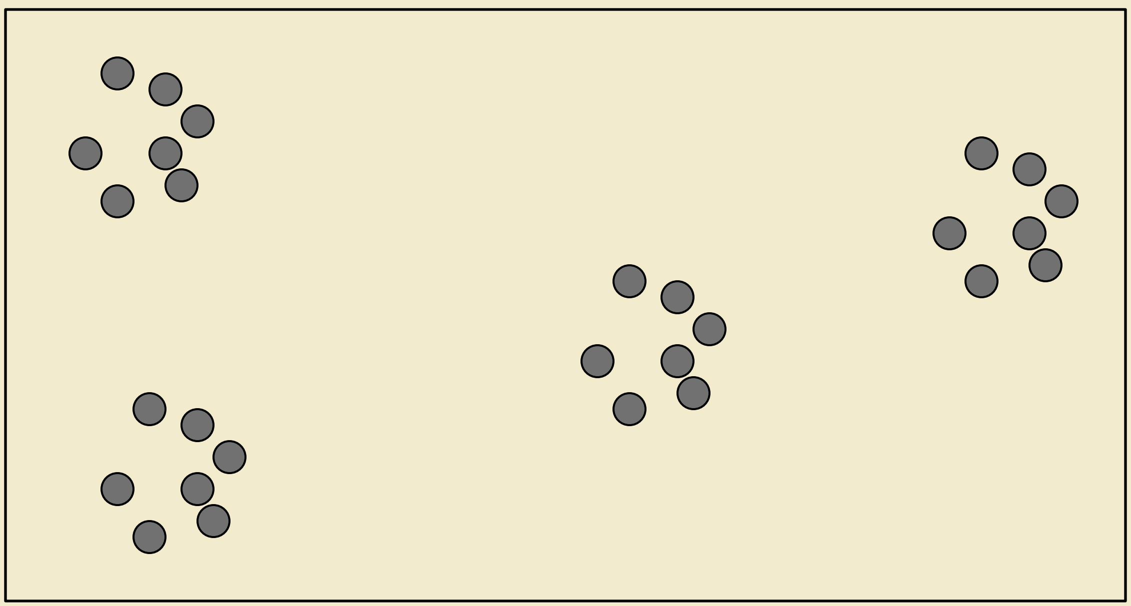
K-means++ [Arthur et al. '07]

- Spreads out the centers
- Choose first center, c_1, uniformly at random from the data set
- Repeat for $2 \leq i \leq k$:
  - Choose c_i to be equal to a data point x_0 sampled from the distribution:

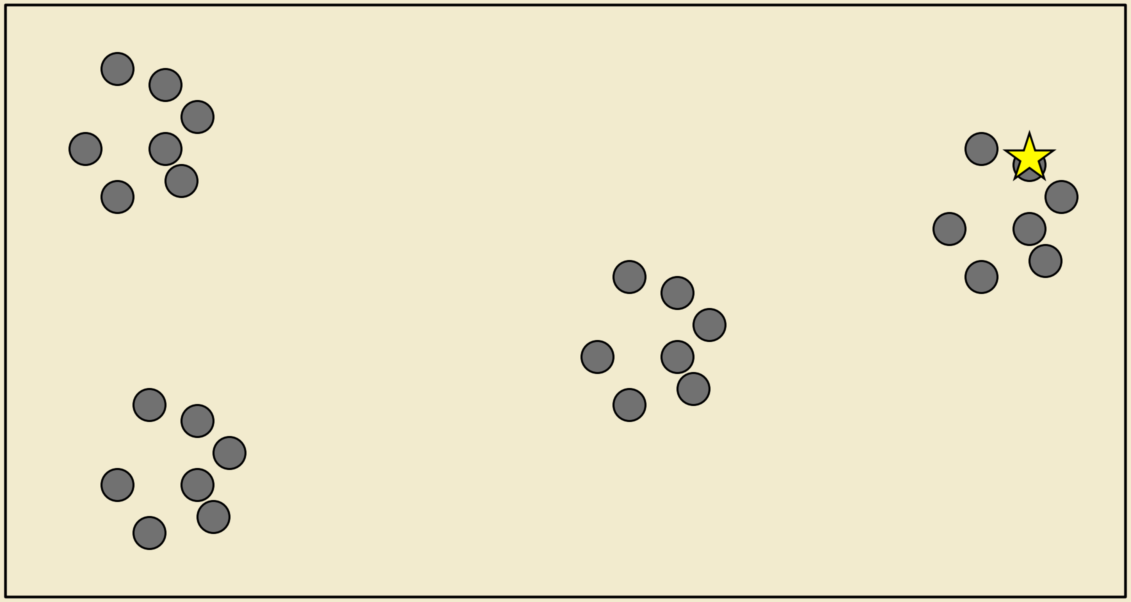
$$\frac{d(x_0, C)^2}{\varphi_X(C)} \propto d(x_0, C)^2$$

- **Theorem:** $O(\log k)$-approximation to optimum, right after initialization

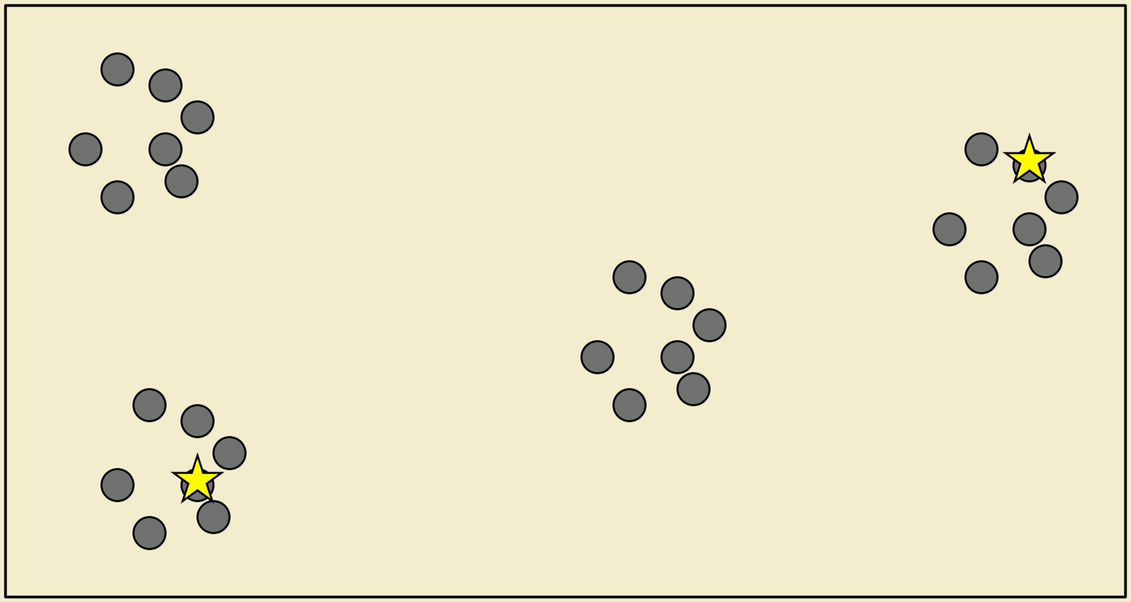
K-means++ Initialization



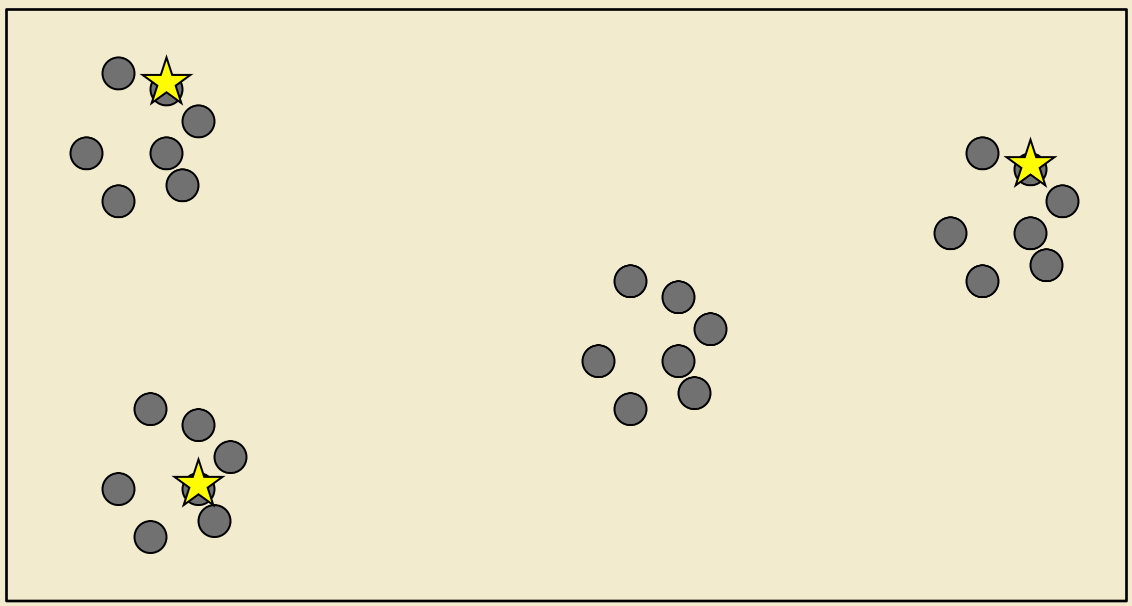
K-means++ Initialization



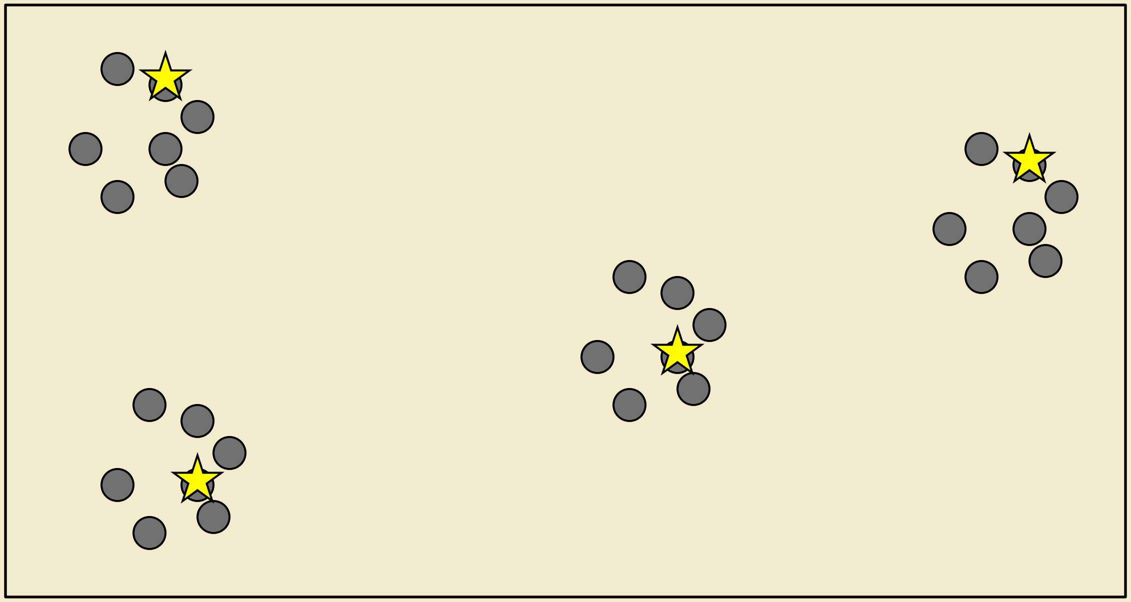
K-means++ Initialization



K-means++ Initialization



K-means++ Initialization



What's wrong with K-means++?

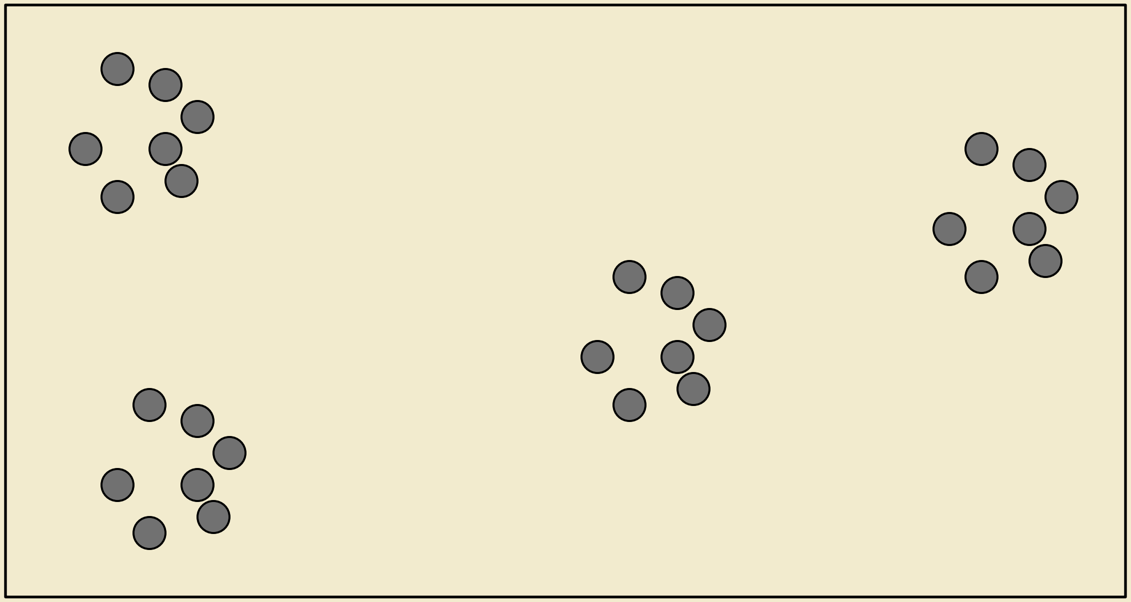
- Needs K passes over the data
- In large data applications, not only the data is massive, but also K is typically large (e.g., easily 1000).
- Does not scale!

Intuition for a solution

- K-means++ samples one point per iteration and updates its distribution
- What if we **oversample** by sampling each point independently with a larger probability?
- Intuitively equivalent to updating the distribution much less frequently
  - Coarser sampling
- Turns out to be sufficient: K-means||

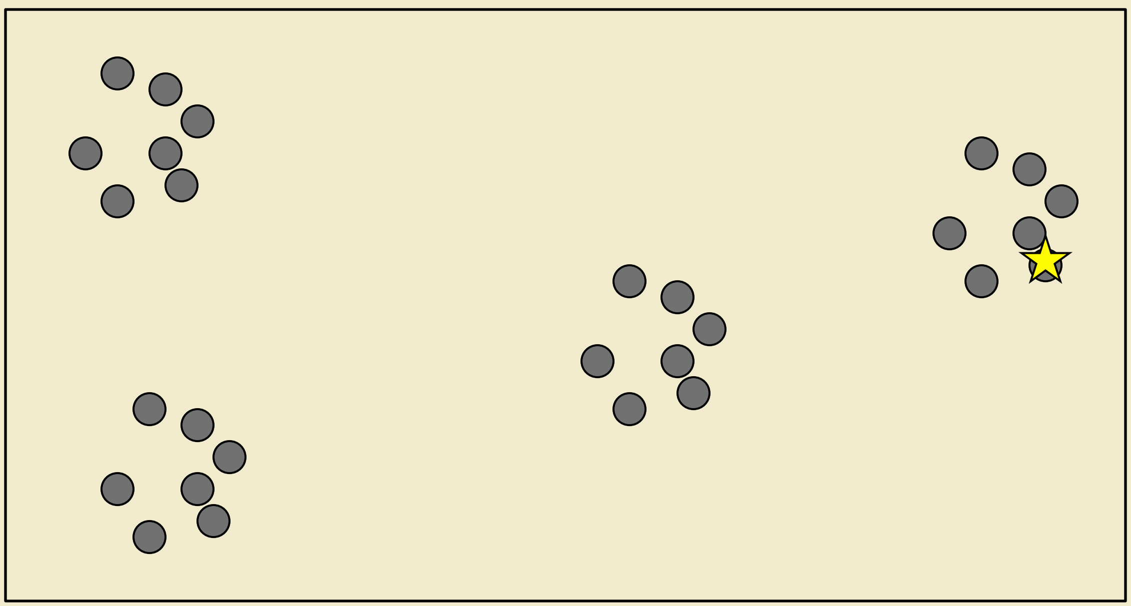
K-means|| Initialization

K=4,
Oversampling factor =3



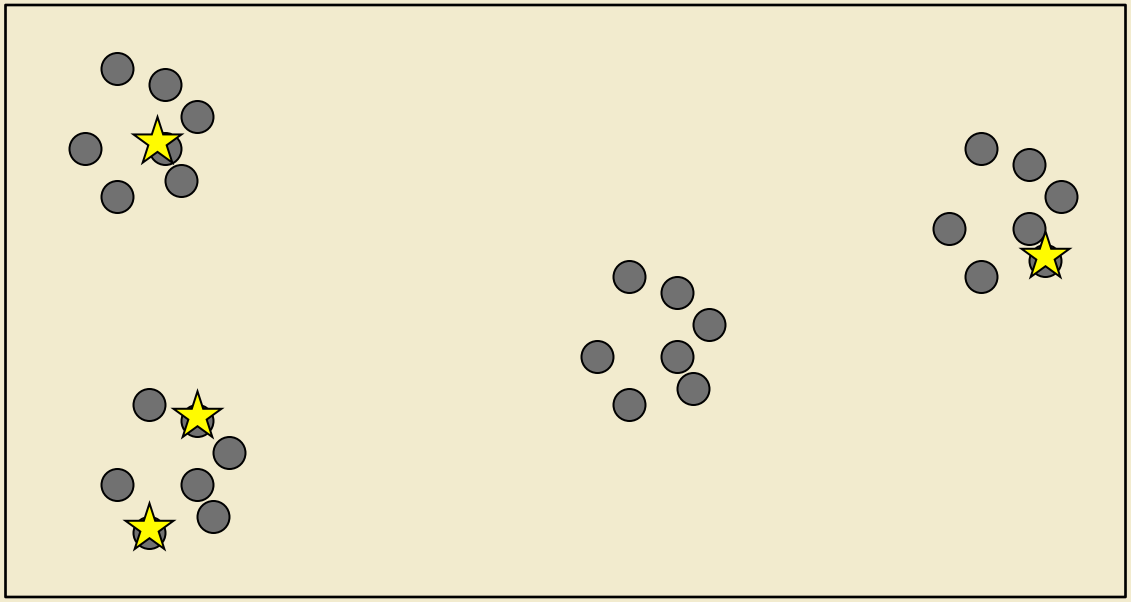
K-means|| Initialization

K=4,
Oversampling factor =3



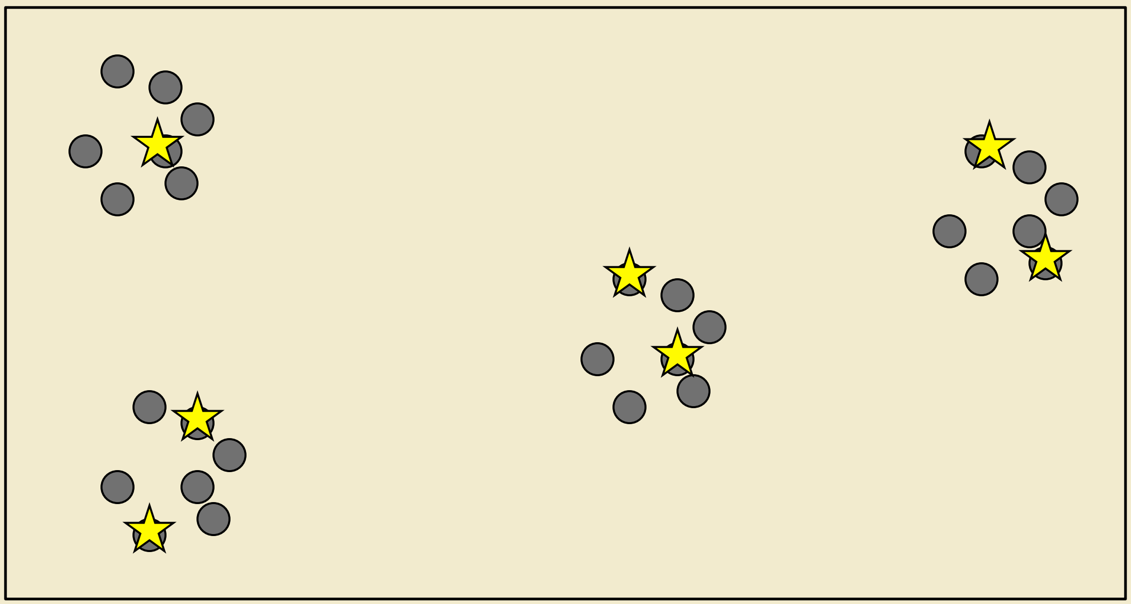
K-means|| Initialization

K=4,
Oversampling factor =3



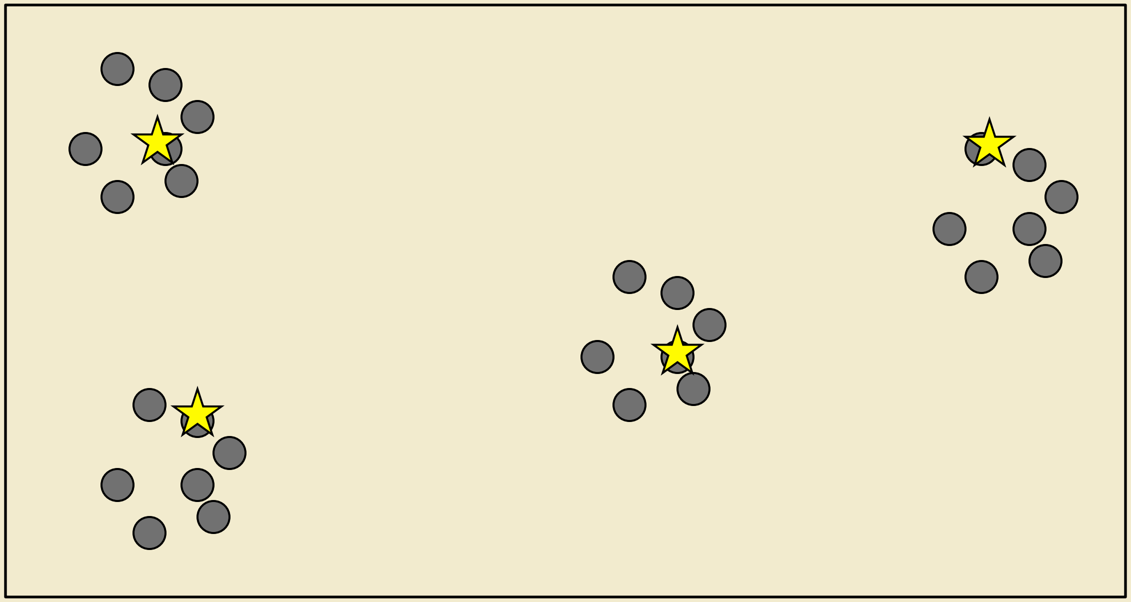
K-means|| Initialization

K=4,
Oversampling factor =3



K-means|| Initialization

K=4,
Oversampling factor =3



Cluster the intermediate centers

K-means|| [Bahmani et al. '12]

- Choose $l>1$ [Think $l=\Theta(k)$]
- Initialize C to an arbitrary set of points
- For R iterations do:
  - Sample each point x in X independently with probability $p_x = l d^2(x,C)/\varphi_X(C)$.
  - Add all the sampled points to C
- Cluster the (weighted) points in C to find the final k centers

K-means||: Intuition

- An interpolation between Lloyd and K-means++

Number of iterations (R)

R=k: Simulating K-means++ (l=1) → Strong guarantee

Small R: K-means|| → **Can it possibly give any guarantees?**

R=0: Lloyd → No guarantees

Theorem

- **Theorem:** If φ and φ' are the costs of the clustering at the beginning and end of an iteration, and OPT is the cost of the optimum clustering:

$$E[\varphi'] \leq O(OPT) + \frac{k}{el}\varphi$$

- **Corollary:**
  - Let ψ = cost of initial clustering
  - K-means|| produces a constant-factor approximation to OPT, using only O(log (ψ /OPT)) iterations
  - Using K-means++ for clustering the intermediate centers, the overall approximation factor = **O(log k)**

Experimental Results: Quality

	Clustering Cost Right After Initialization	Clustering Cost After Lloyd Convergence
Random	NA	22,000
K-means++	430	65
K-means||	16	14

GAUSSMIXTURE: 10,000 points in 15 dimensions
K=50
Costs scaled down by 10^4

- K-means|| much harder than K-means++ to get confused with noisy outliers

Experimental Results: Convergence

- K-means|| reduces number of Lloyd iterations even more than K-means++

	Number of Lloyd Iterations till Convergence
Random	167
K-means++	42
K-means||	28

SPAM: 4,601 points in 58 dimensions
K=50

Experimental Results

- K-means|| needs a small number of intermediate centers
- Better than K-means++ as soon as ~K centers chosen

	Clustering Cost (Scaled down by 10^{10})	Number of intermediate centers	Tme (In Minutes)
Random	$6.4 * 10^7$	NA	489
Partition	1.9	$1.47 * 10^6$	1022
K-means||	1.5	3604	87

KDDCUP1999: 4.8M points in 42 dimensions
K=1000

Algorithmic Theme

- Quickly decrease the size of the data in a distributed fashion…
- … while maintaining the important features of the data
- Solve the small instance on a single machine

Thank You!