Simple and sharp analysis of *k*-means||

Vasek Rozhon
ETH, Zurich

Plan

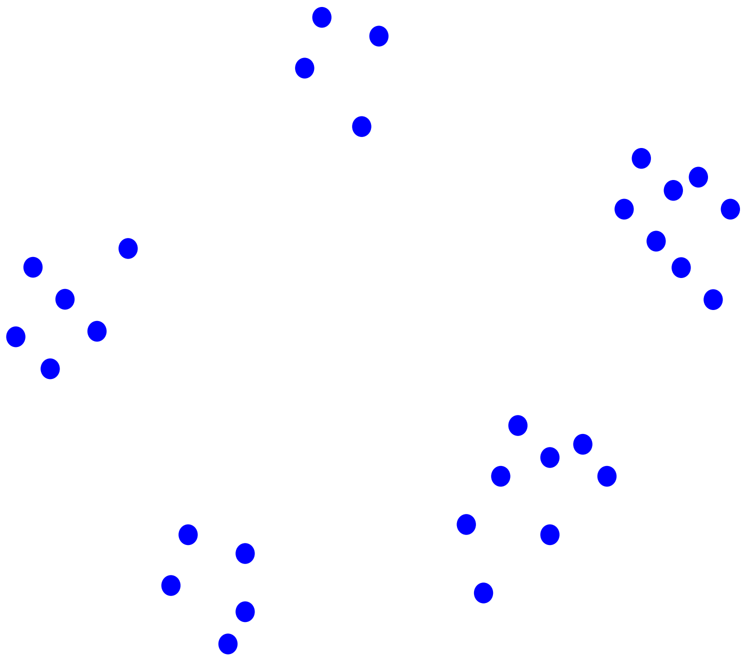
1) define the k-means problem
2) talk about k-means++
3) see a simple analysis of the distributed version of k-means++ (called k-means||)

Plan

1) define the k-means problem

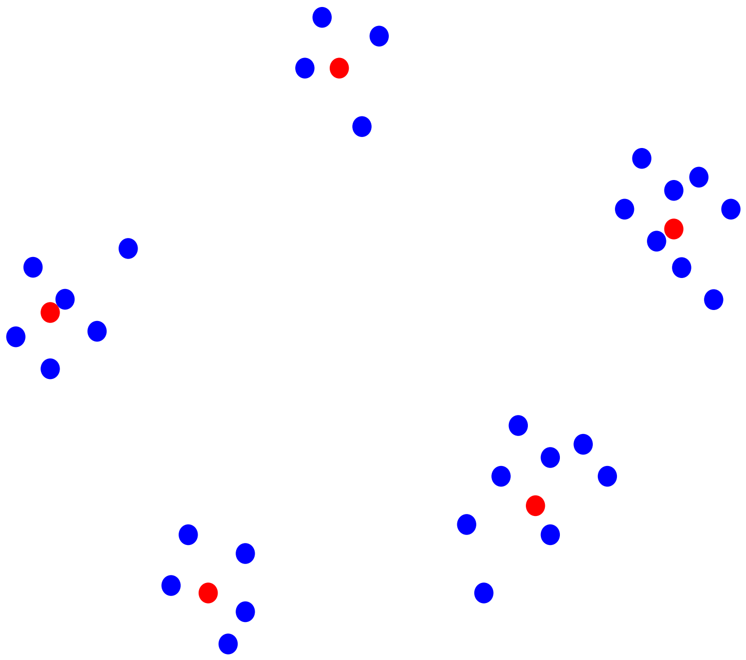
k-means: definition

For a set X find a set of k centers C that minimizes $\Sigma_{x \in X} \min_{c \in C} d(x,c)^2$



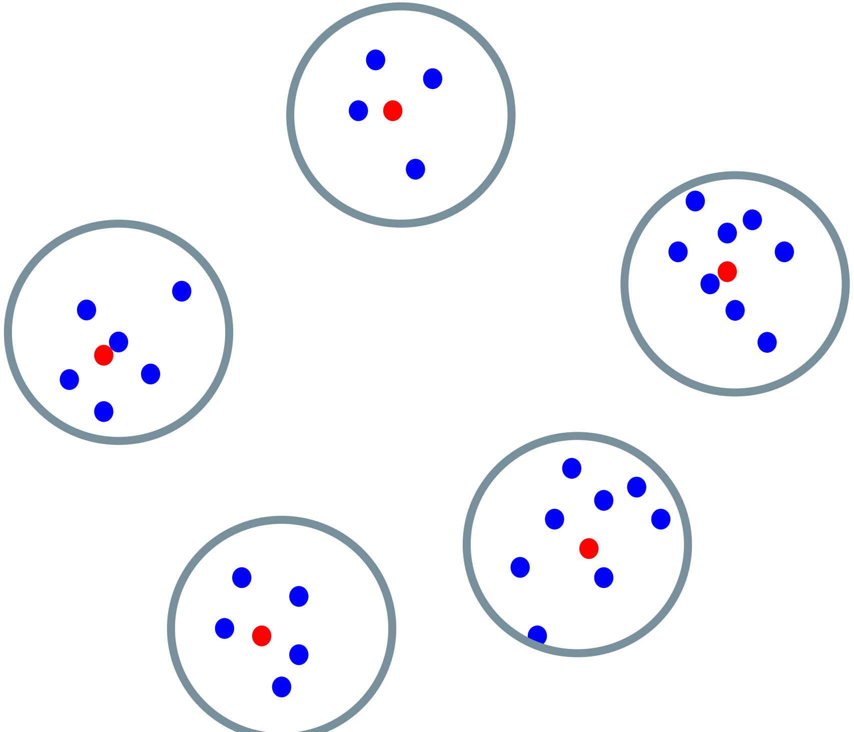
k-means: definition

For a set X find a set of k centers C that minimizes $\Sigma_{x \in X} \min_{c \in C} d(x,c)^2$

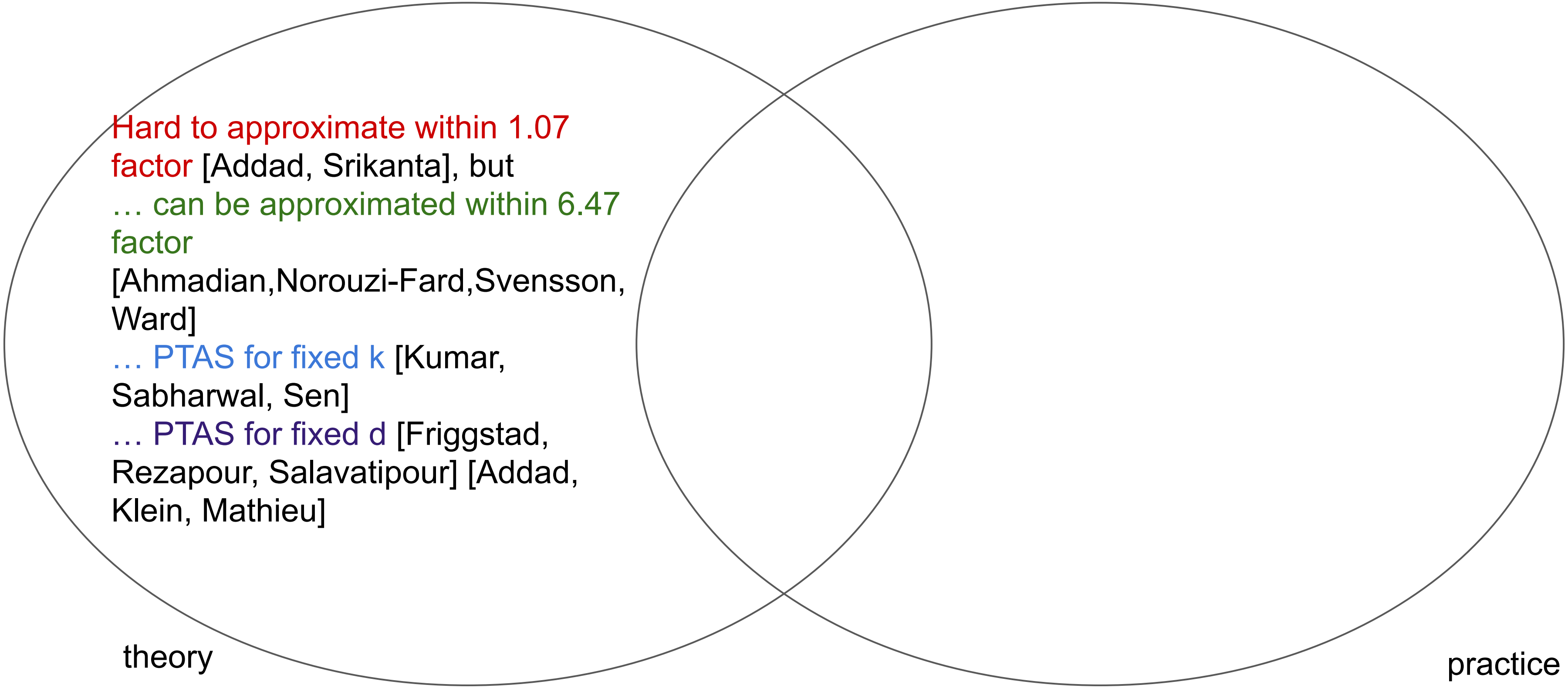


k-means: definition

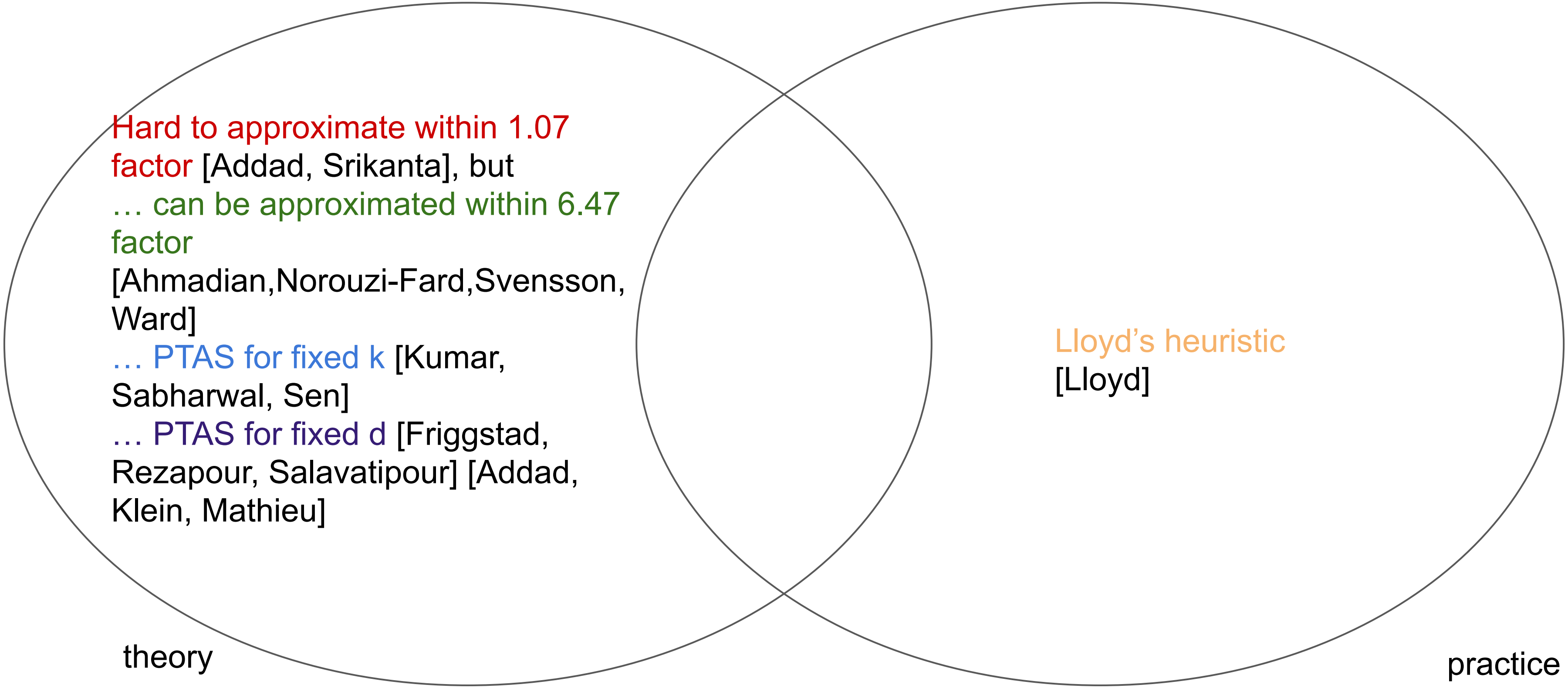
For a set X find a set of k centers C that minimizes $\Sigma_{x \in X} \min_{c \in C} d(x,c)^2$



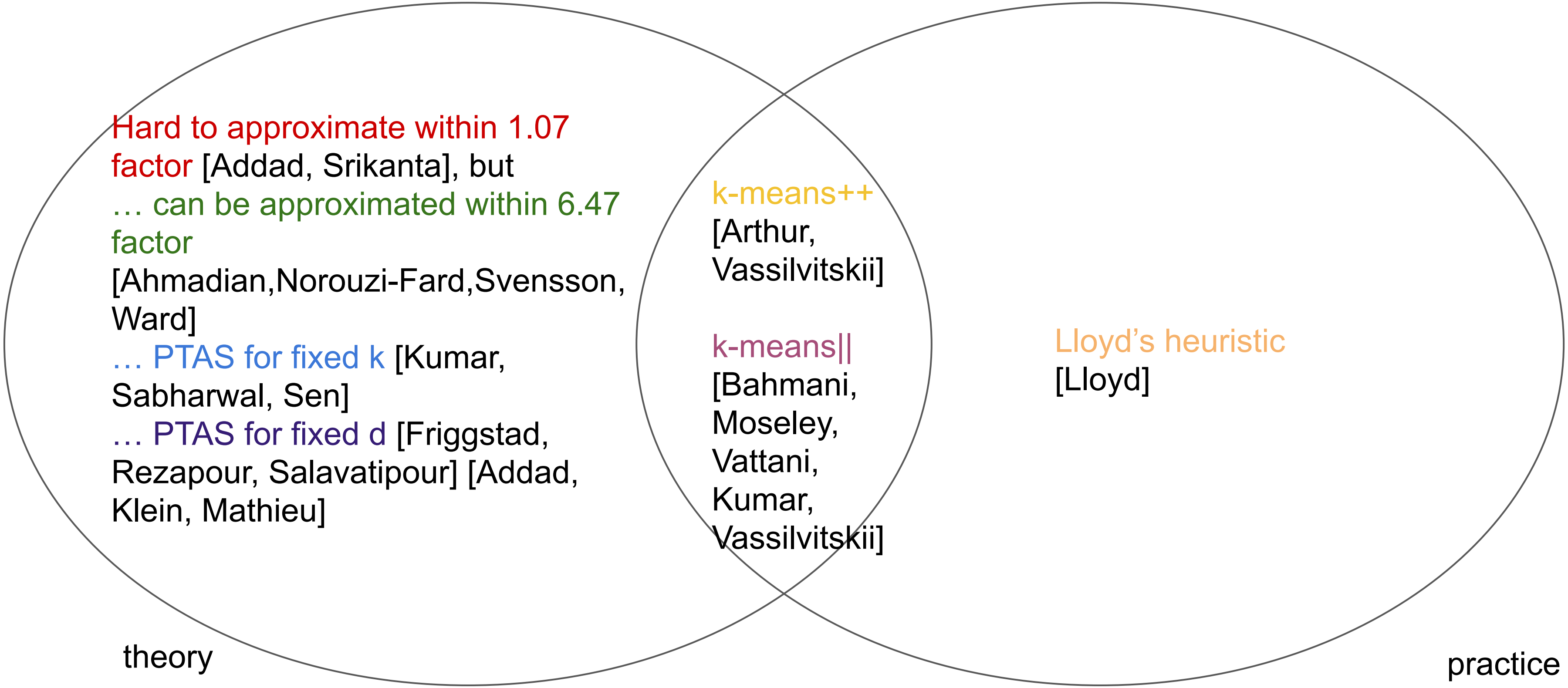
k-means: theory versus practice



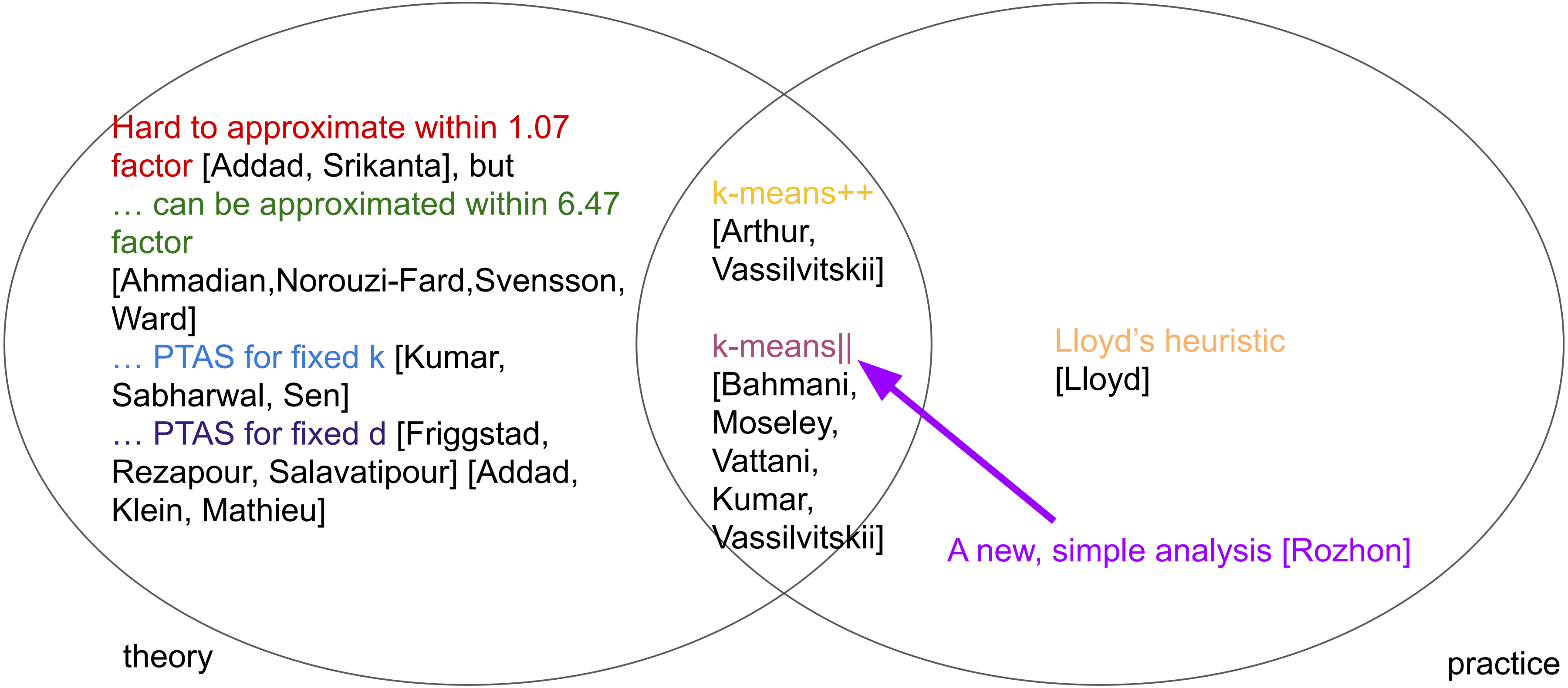
k-means: theory versus practice



k-means: theory versus practice



k-means: theory versus practice



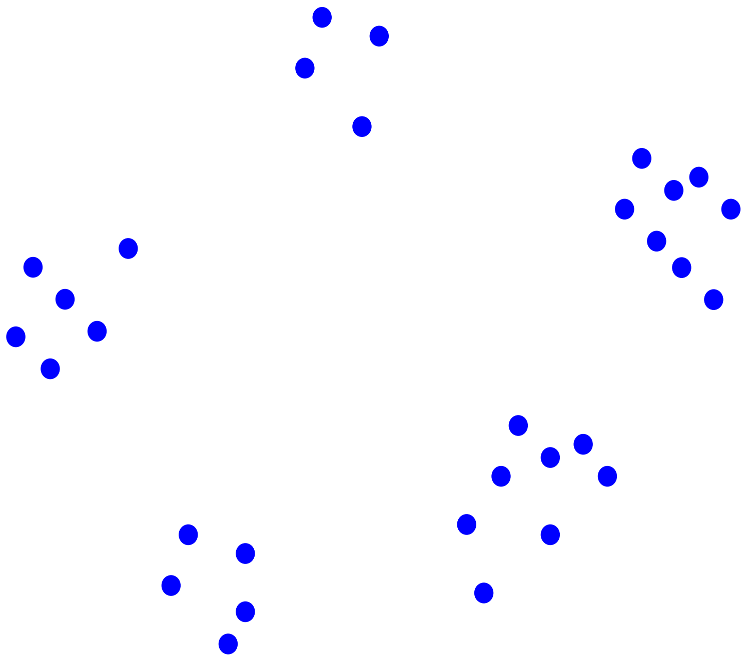
Plan

1) define the k-means problem
2) talk about k-means++

k-means++

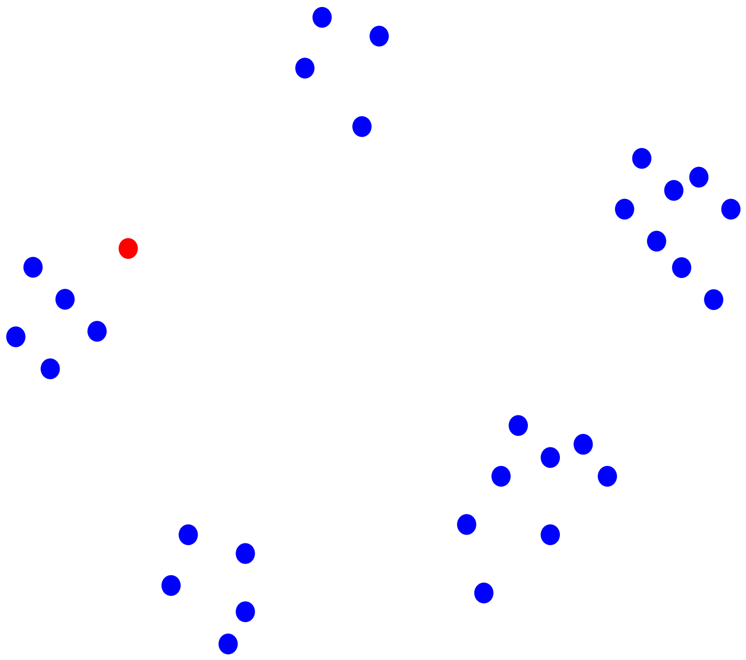
Practice: fast seeding for Lloyd’s algorithm

Theory: expected O(log k) approximation guarantee

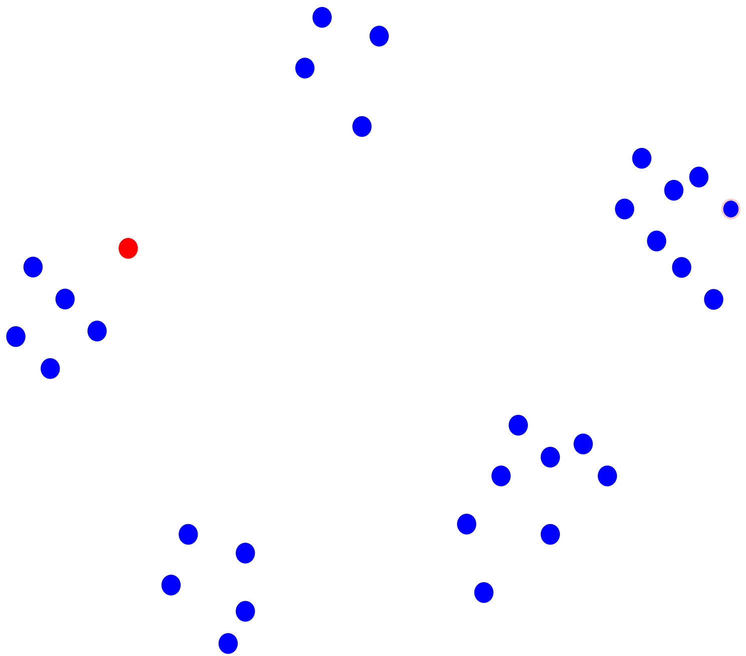


k-means++

First center: uniformly at random



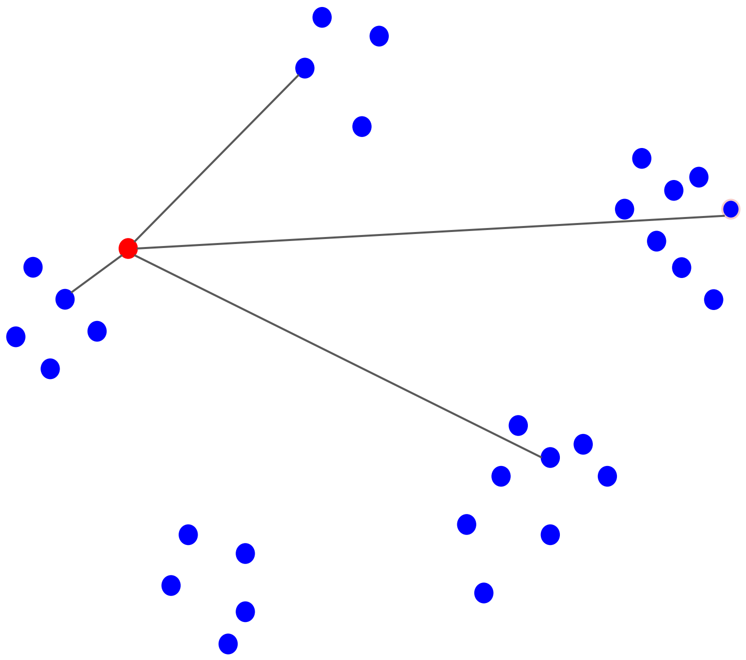
k-means++



First center: uniformly at random

Next k-1 centers: sample a point proportional to its current cost

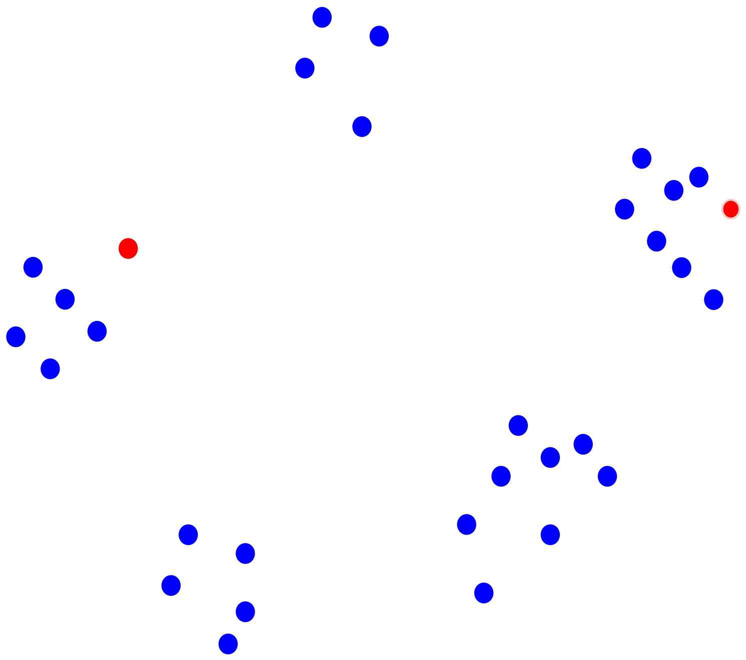
k-means++



First center: uniformly at random

Next k-1 centers: sample a point proportional to its current cost

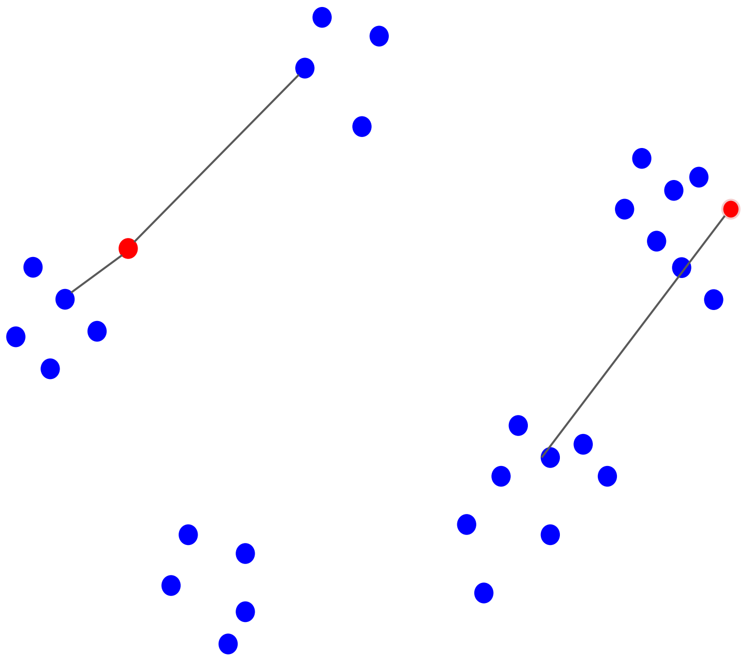
k-means++



First center: uniformly at random

Next k-1 centers: sample a point proportional to its current cost

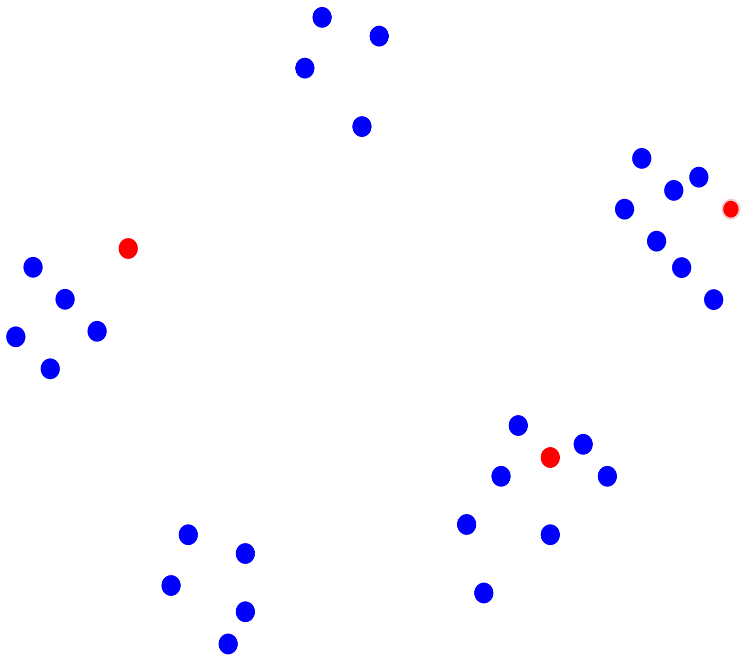
k-means++



First center: uniformly at random

Next k-1 centers: sample a point proportional to its current cost

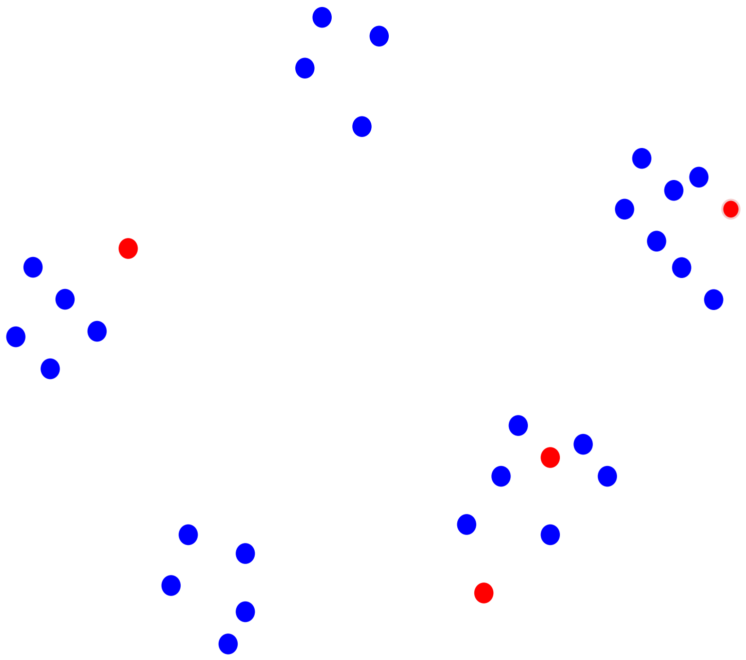
k-means++



First center: uniformly at random

Next k-1 centers: sample a point proportional to its current cost

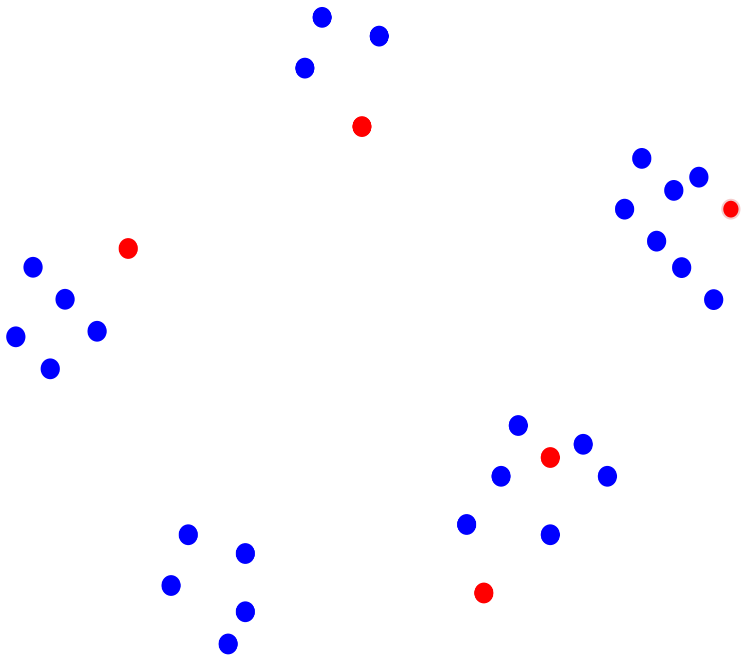
k-means++



First center: uniformly at random

Next k-1 centers: sample a point proportional to its current cost

k-means++



First center: uniformly at random

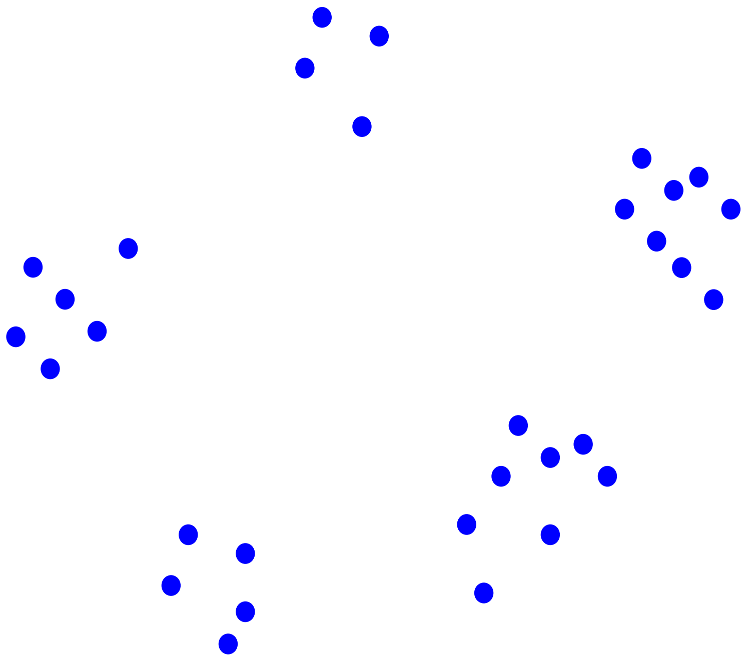
Next k-1 centers: sample a point proportional to its current cost

Plan

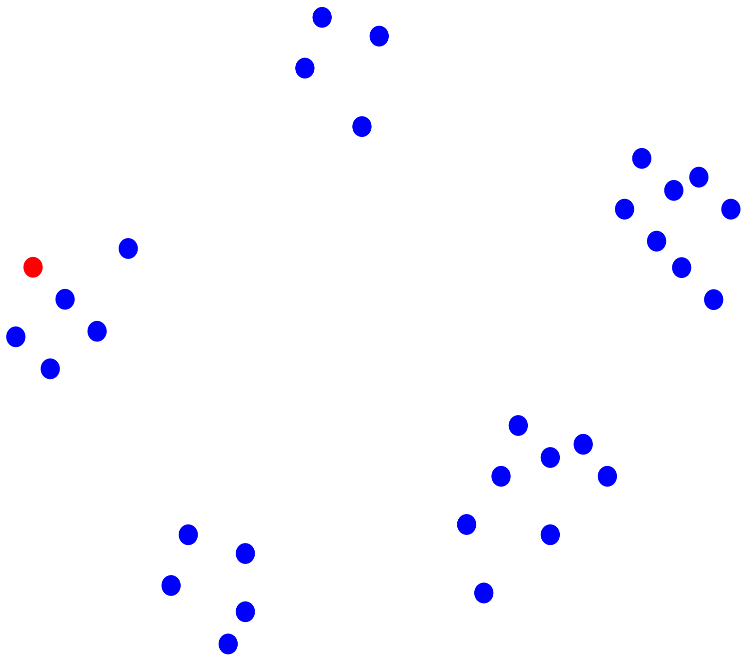
1) define the k-means problem
2) talk about k-means++
3) see a simple analysis of the distributed version of k-means++ (called k-means||)

k-means|| [Bahmani, Moseley, Vattani, Kumar, Vassilvitskii]

Distributed (e.g. MapReduce) variant of k-means++



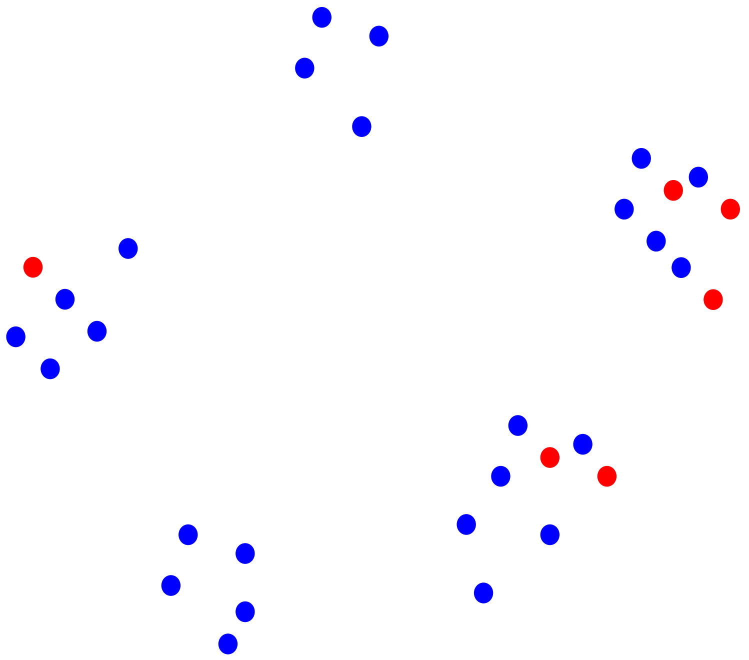
k-means|| [Bahmani, Moseley, Vattani, Kumar, Vassilvitskii]



Distributed (e.g. MapReduce) variant of k-means++

First point sampled at random.

k-means|| [Bahmani, Moseley, Vattani, Kumar, Vassilvitskii]

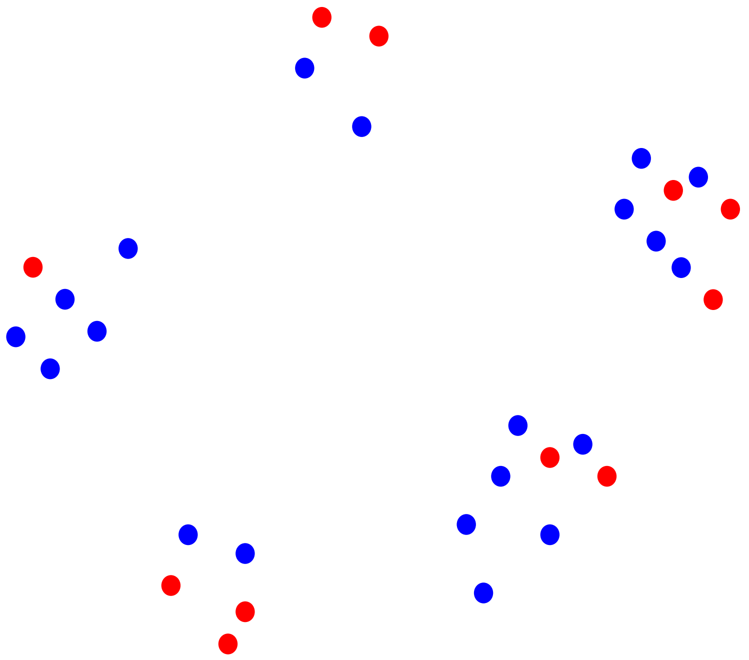


Distributed (e.g. MapReduce) variant of k-means++

First point sampled at random.

Next, sample k points in parallel proportional to cost.

k-means|| [Bahmani, Moseley, Vattani, Kumar, Vassilvitskii]



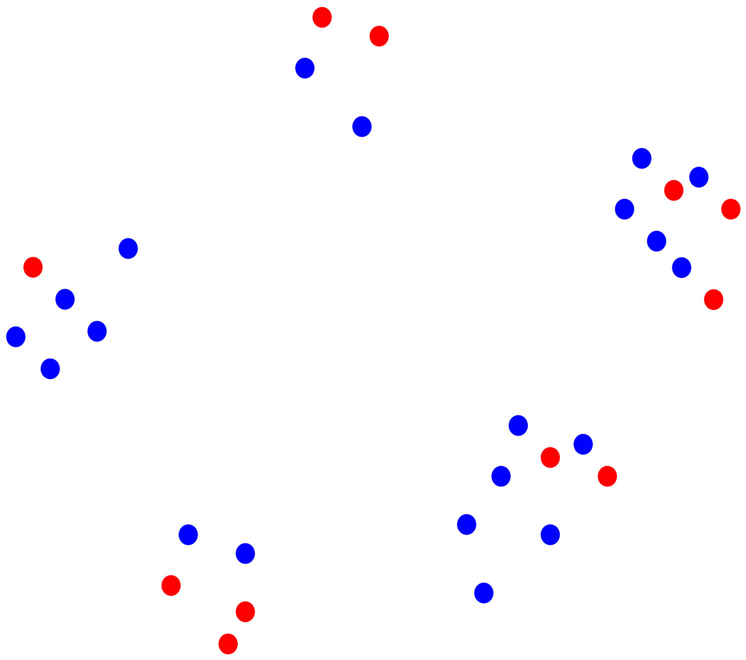
Distributed (e.g. MapReduce) variant of k-means++

First point sampled at random.

Next, sample k points in parallel proportional to cost.

Next, sample k points in parallel proportional to cost.

k-means|| [Bahmani, Moseley, Vattani, Kumar, Vassilvitskii]



Distributed (e.g. MapReduce) variant of k-means++

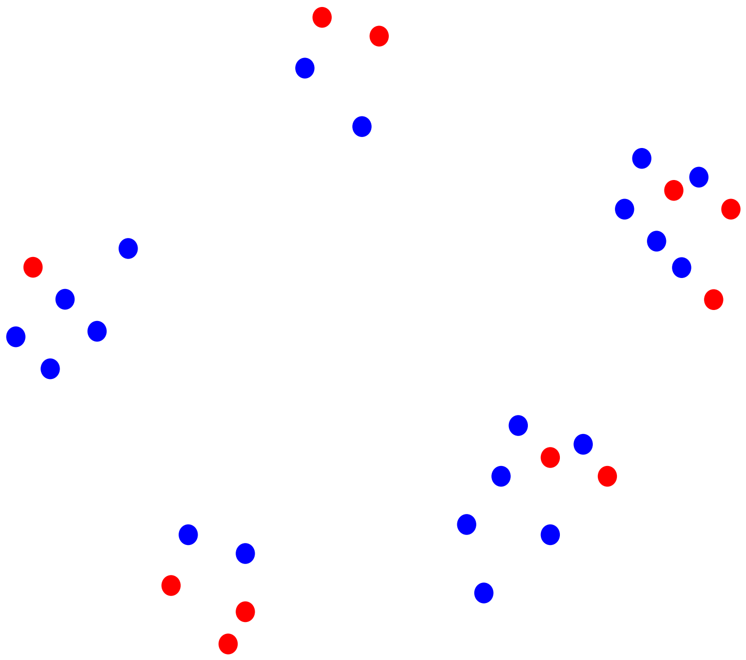
First point sampled at random.

Next, sample k points in parallel proportional to cost.

Next, sample k points in parallel proportional to cost.

Continue for a while.

k-means|| [Bahmani, Moseley, Vattani, Kumar, Vassilvitskii]



Distributed (e.g. MapReduce) variant of k-means++

First point sampled at random.

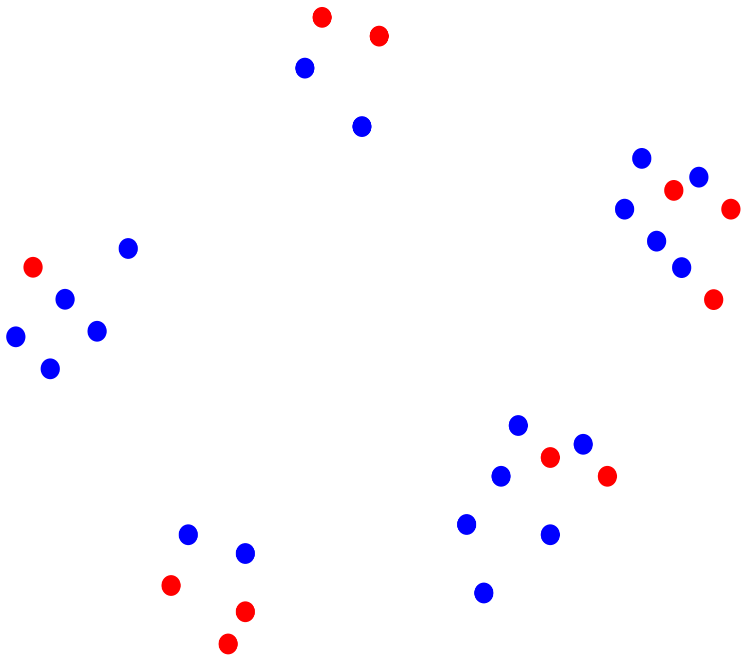
Next, sample k points in parallel proportional to cost.

Next, sample k points in parallel proportional to cost.

Continue for a while.

There is a simple trick that compresses the number of centers back to k.

k-means|| [Bahmani, Moseley, Vattani, Kumar, Vassilvitskii]



Distributed (e.g. MapReduce) variant of k-means++

First point sampled at random.

Next, sample k points in parallel proportional to cost.

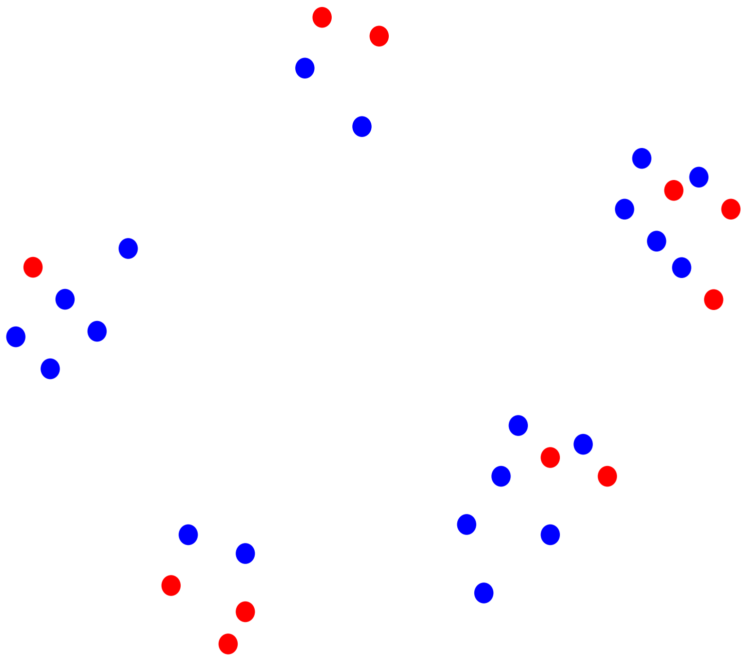
Next, sample k points in parallel proportional to cost.

Continue for a while.

There is a simple trick that compresses the number of centers back to k.

Question: how many steps are needed to get O(1) approximation of the optimum cost?

k-means|| [Bahmani, Moseley, Vattani, Kumar, Vassilvitskii]



Distributed (e.g. MapReduce) variant of k-means++

First point sampled at random.

Next, sample k points in parallel proportional to cost.

Next, sample k points in parallel proportional to cost.

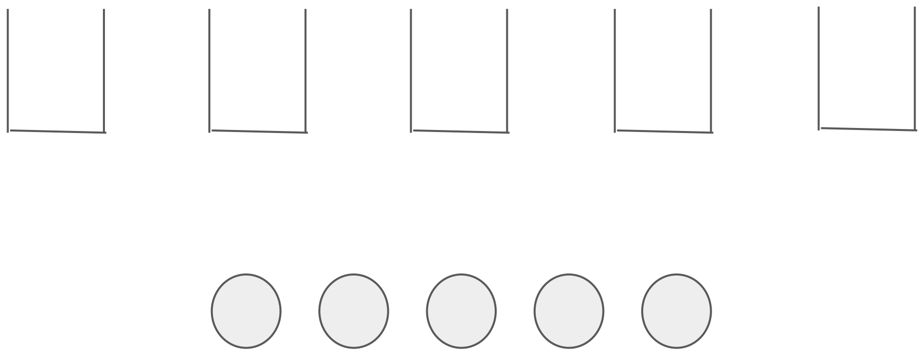
Continue for a while.

There is a simple trick that compresses the number of centers back to k.

Question: how many steps are needed to get O(1) approximation of the optimum cost?

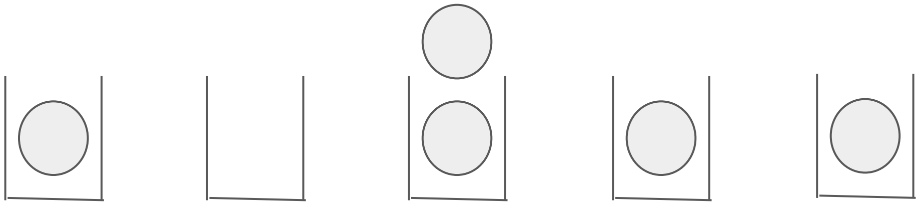
Answer [Bahmani et al., Bachem et al., Rozhon]:
O(log n) steps suffice

Balls into bins



Throw k balls into k bins, each ball to a uniformly random bin.

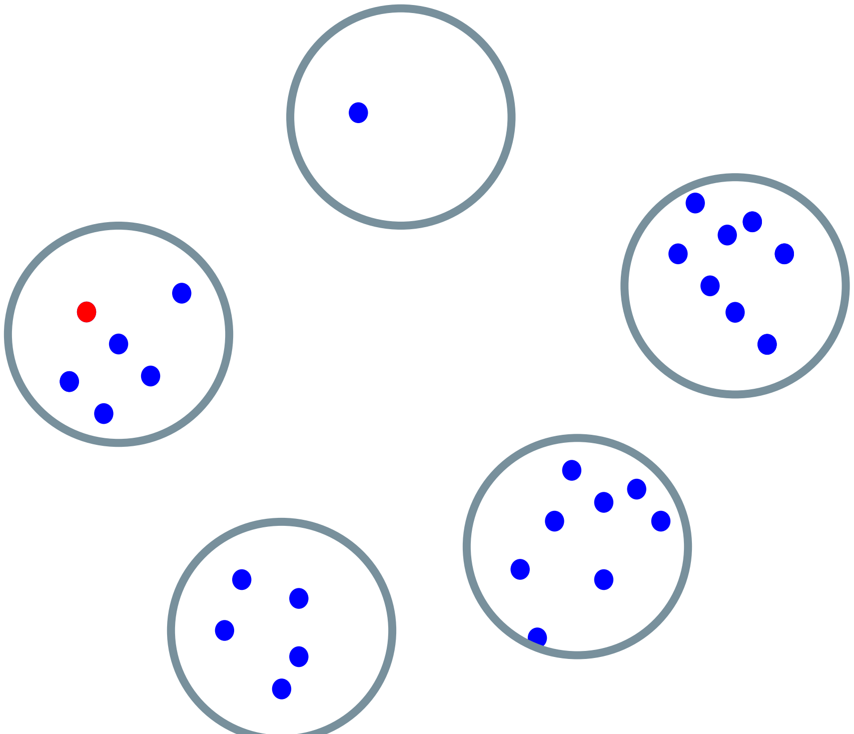
Balls into bins



Each bin is hit with probability $1 - (1 - 1/k)^k \approx 1 - 1/e$.

Hence, we expect to hit a constant fraction of bins.

k-means||: our analysis



One step of k-means|| is just a weighted version of balls into bins.
Ball = Sampled center
Bin = Cluster

Weight here is the cost of each cluster.

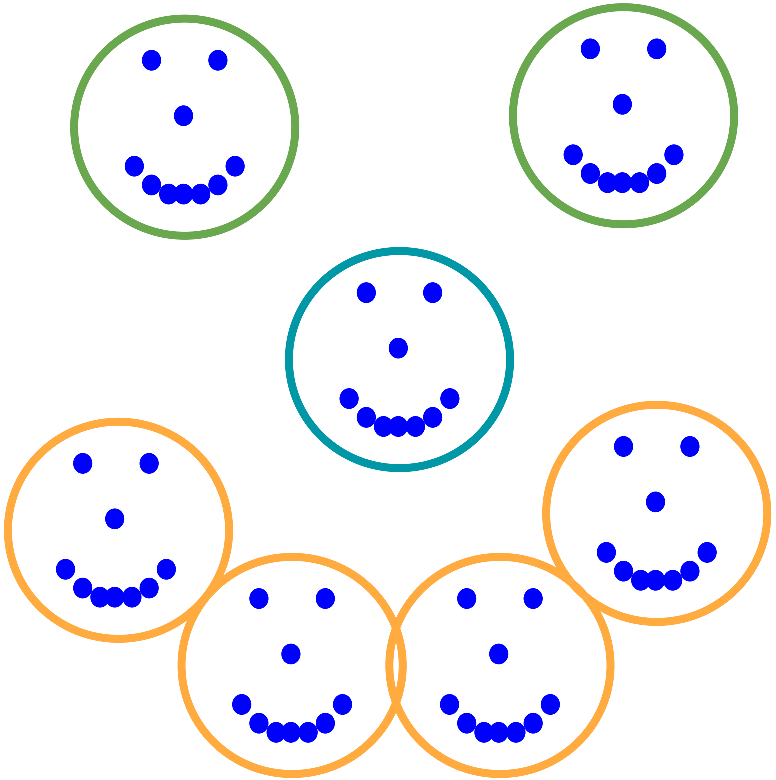
As in classical balls into bins, we expect the total weight to decrease by constant factor in each step.

Hence, O(log n) steps suffice.

Final surprise: this approach can be used to improve the number of rounds needed to O(log n / loglog n).

We show this improved analysis to be tight by extending a lower bound from [Bachem et al.]

k-means||: our analysis



One step of k-means|| is just a weighted version of balls into bins.
Ball = Sampled center
Bin = Cluster

Weight here is the cost of each cluster.

As in classical balls into bins, we expect the total weight to decrease by constant factor in each step.

Hence, O(log n) steps suffice.

Final surprise: this approach can be used to improve the number of rounds needed to O(log n / loglog n).

We show this improved analysis to be tight by extending a lower bound from [Bachem et al.]