TDigest

康凯森

美团点评

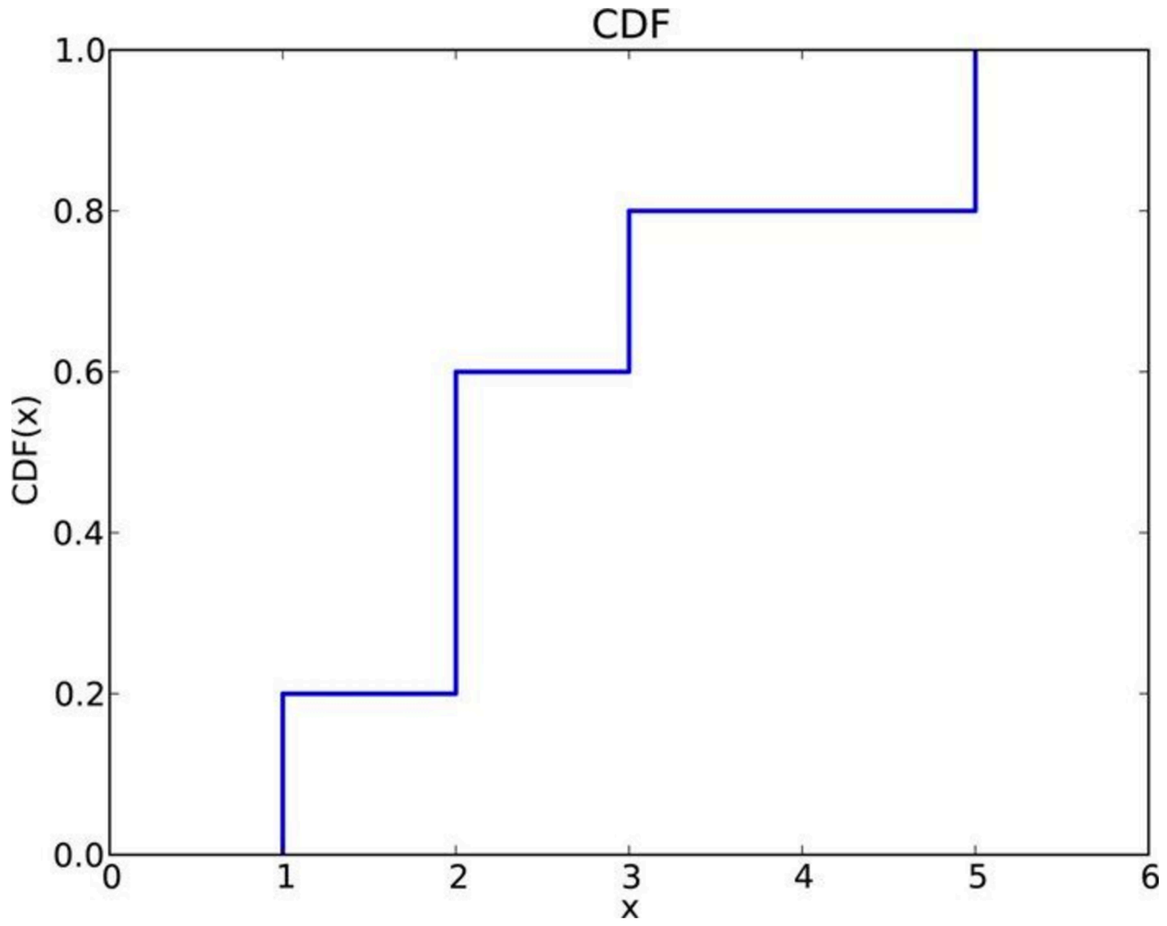
数据平台中心 | 离线计算组

TDigest

Cumulative distribution function

$$F_X(x) = \mathrm{P}(X \leq x)$$

- Example: [1, 2, 2, 3, 5]
  - CDF(0) = 0
  - CDF(2) = 0.6
  - CDF(3) = 0.8
  - CDF(4) = 0.8



Quantile Function

- Inverse cumulative distribution function
- Example: [1, 2, 3, 4, 5]
  - quantile(0.8) = 4 (**80% data less than or equal to 4**)
  - quantile(0.7) = ?

Precisely Compute

Quantile Function

Precisely Compute Sample Code

Quantile Function

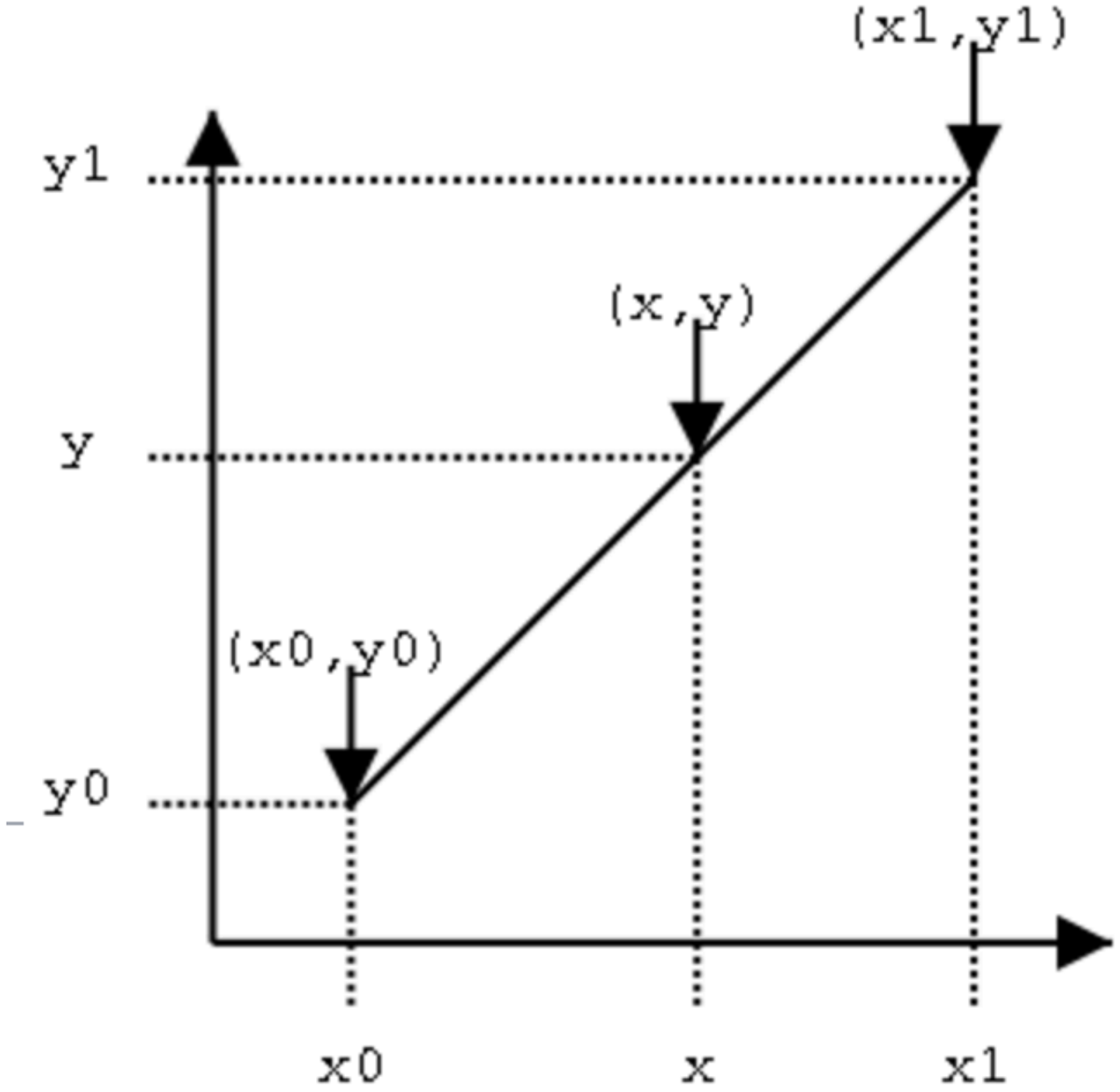
```
quantile(double q, List<Double> data) {
    sort(data); //1 sort

    double index = q * data.size();  //2 index
    int intIndex = (int) index;

    data[intIndex + 1] * (index - intIndex) +
    data[intIndex] * (intIndex + 1 - index);  //3 linear interpolation

    //quantile(0.7) = 3*0.5 + 4*0.5 = 3.5
}
```

Linear Interpolation



The drawback of sorting all data

- Big Data

- Stream Processing

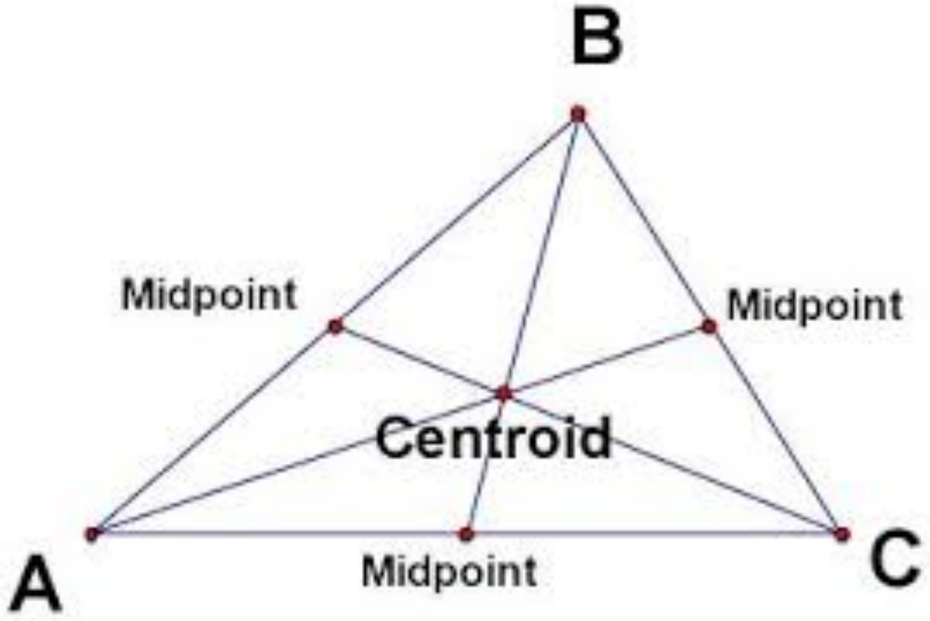
We need sketch

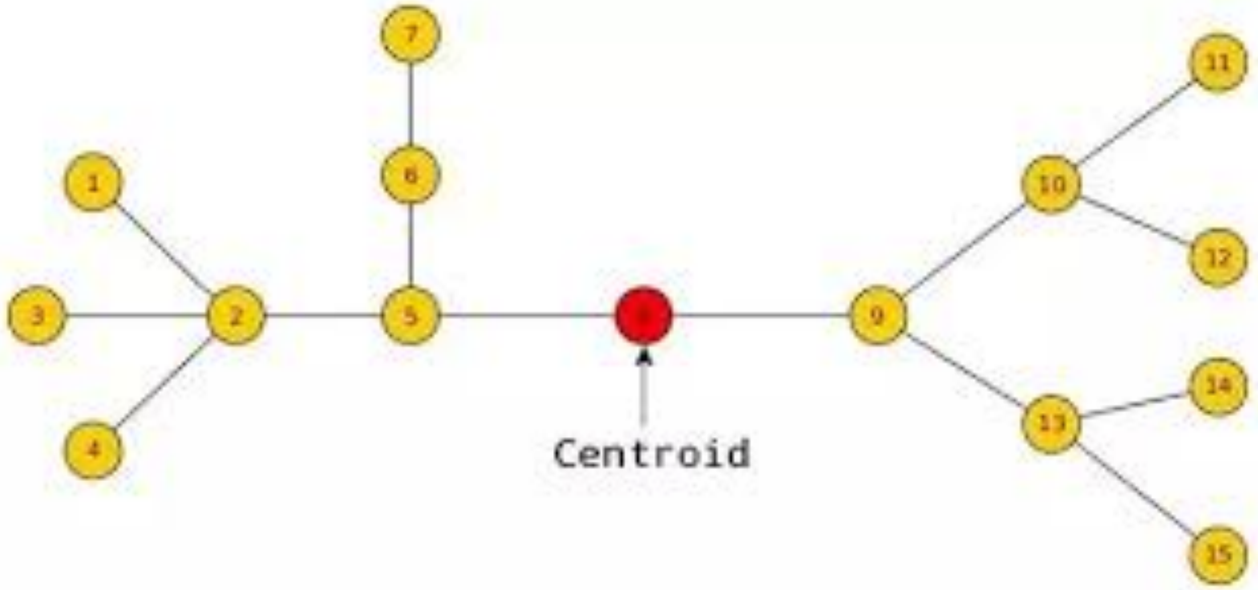
Sketch is common

- Statistics
- Physics
- Deep Learning
- Personas

Centroid

Centroid represents a cluster of points



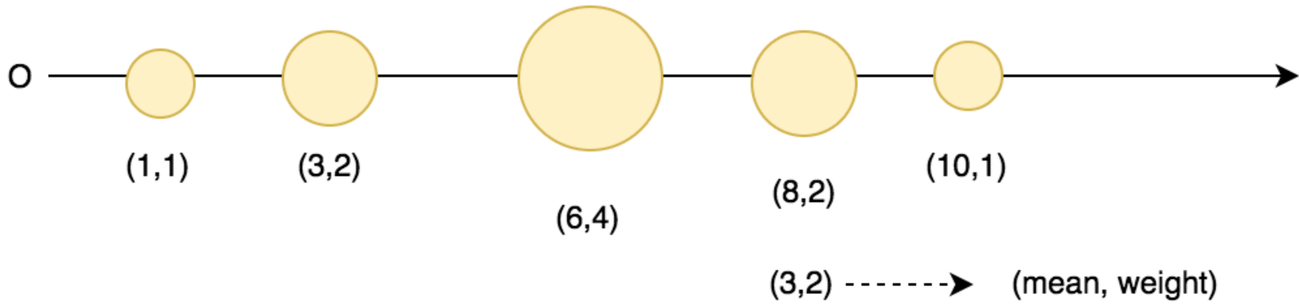


Centroid (mean, weight)

A: [(1,1),(1,1),(1,1),(2,1),(3,1)]

B: [(1,3),(2,1),(3,1)]

(1,1),(1,1),(1,1) ⟶ (1,3)

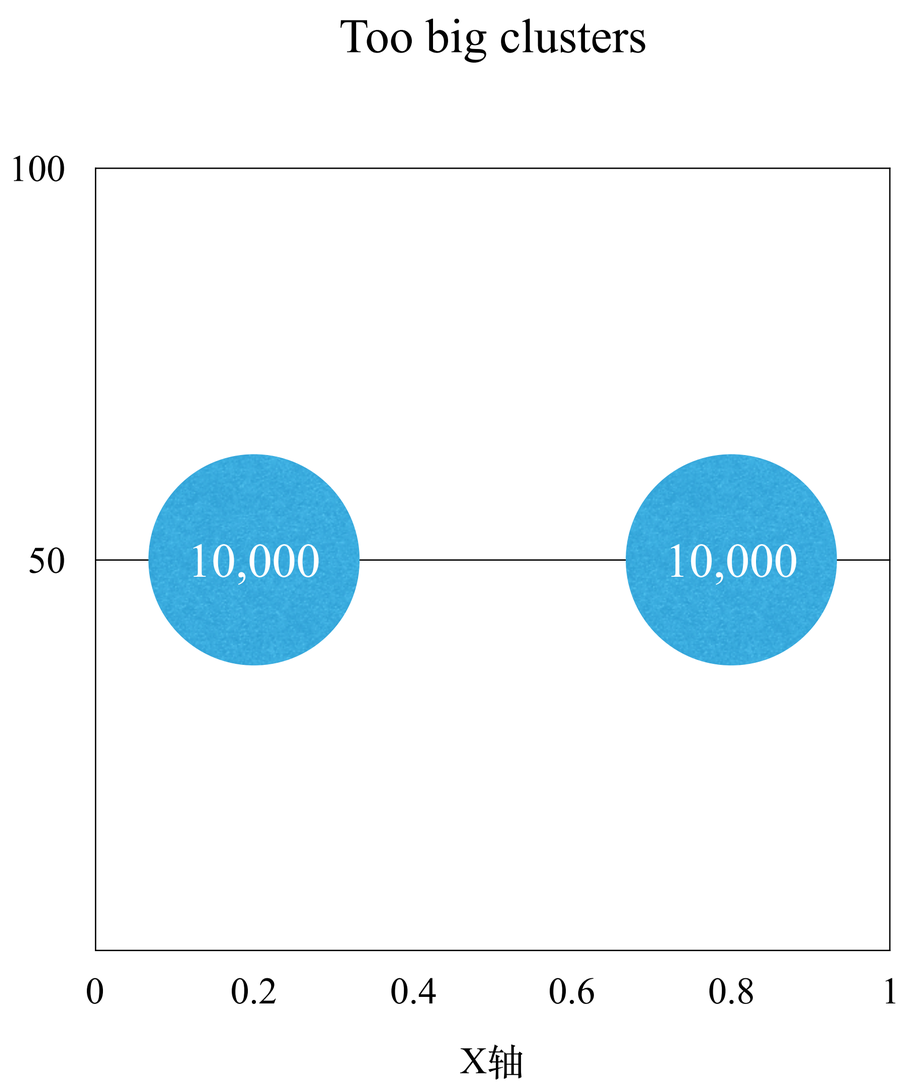


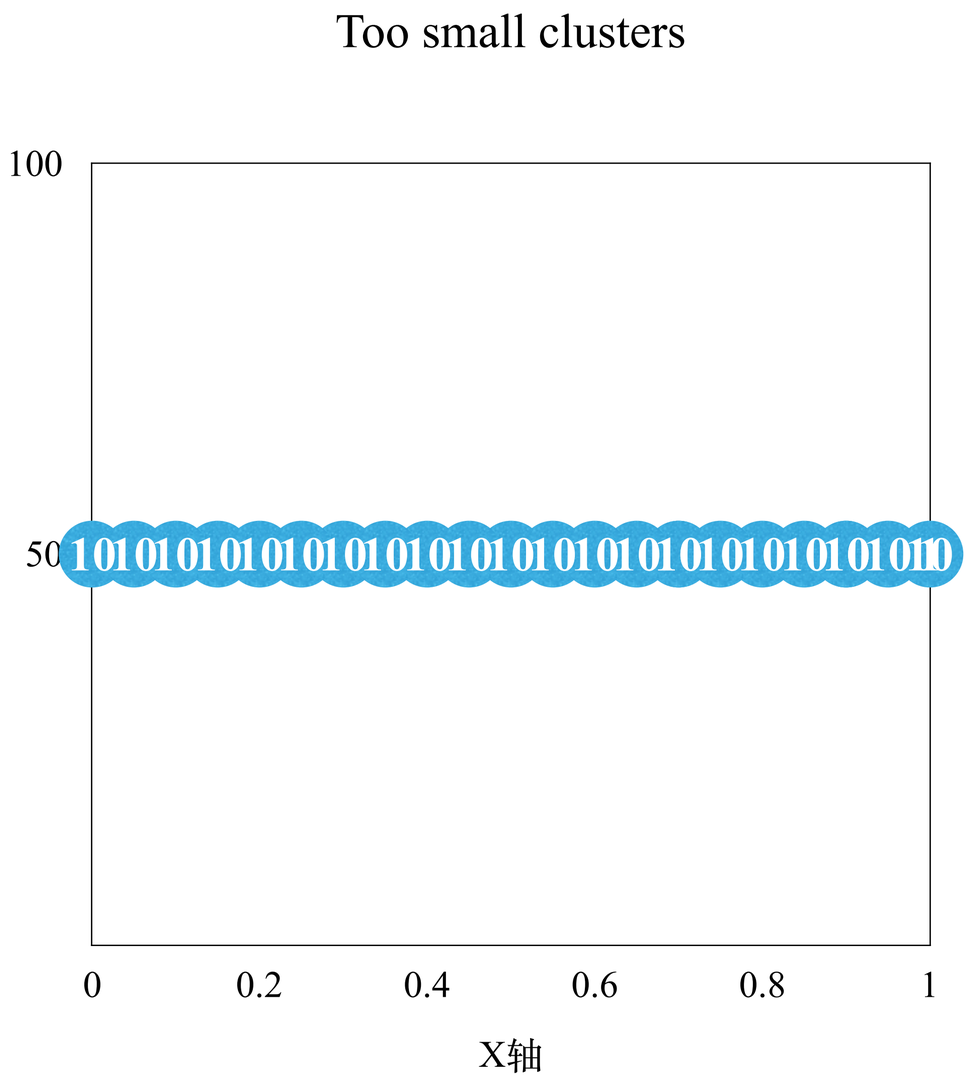
The quantile for A and B is the same

TDigest

How to compute quantile from centroids

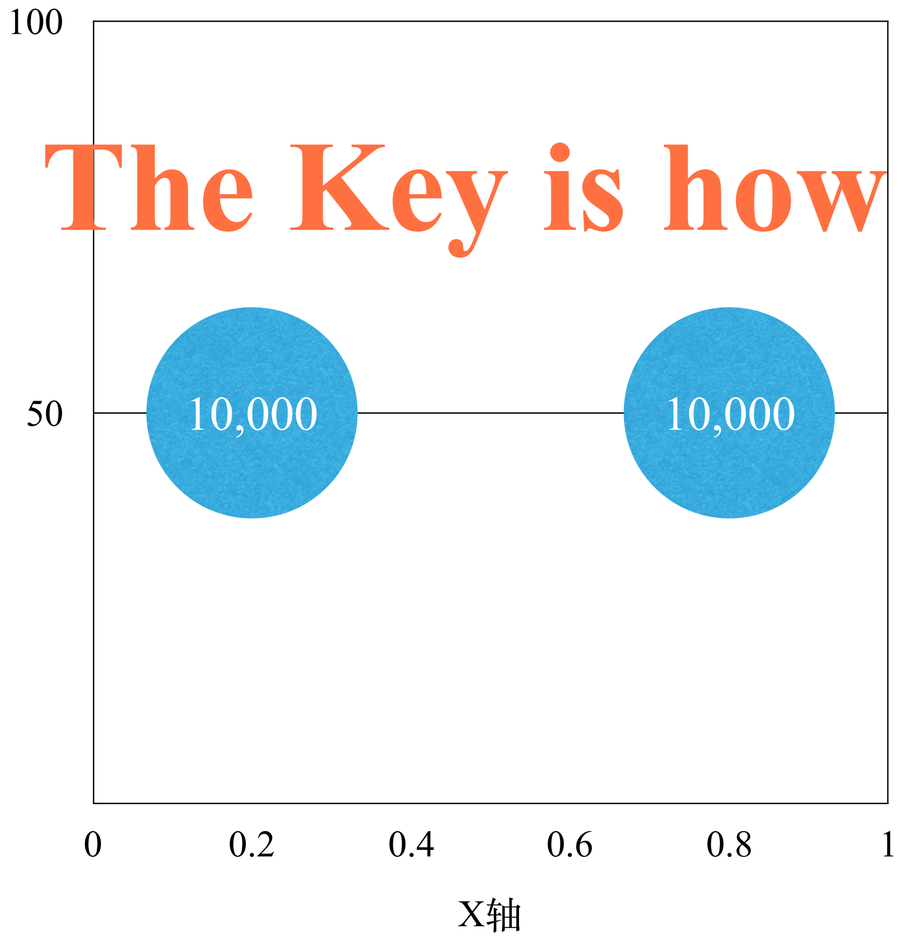
How to compute quantile from centroids



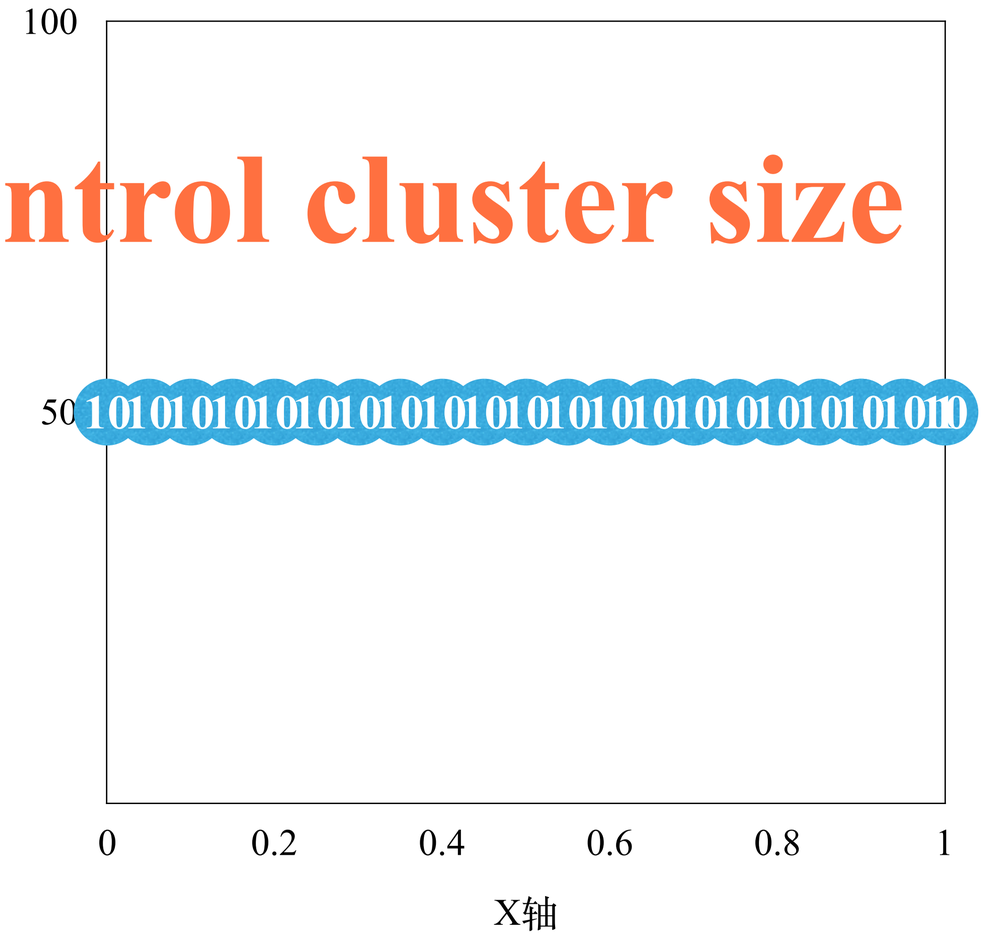


How to compute quantile from centroids

Too big clusters

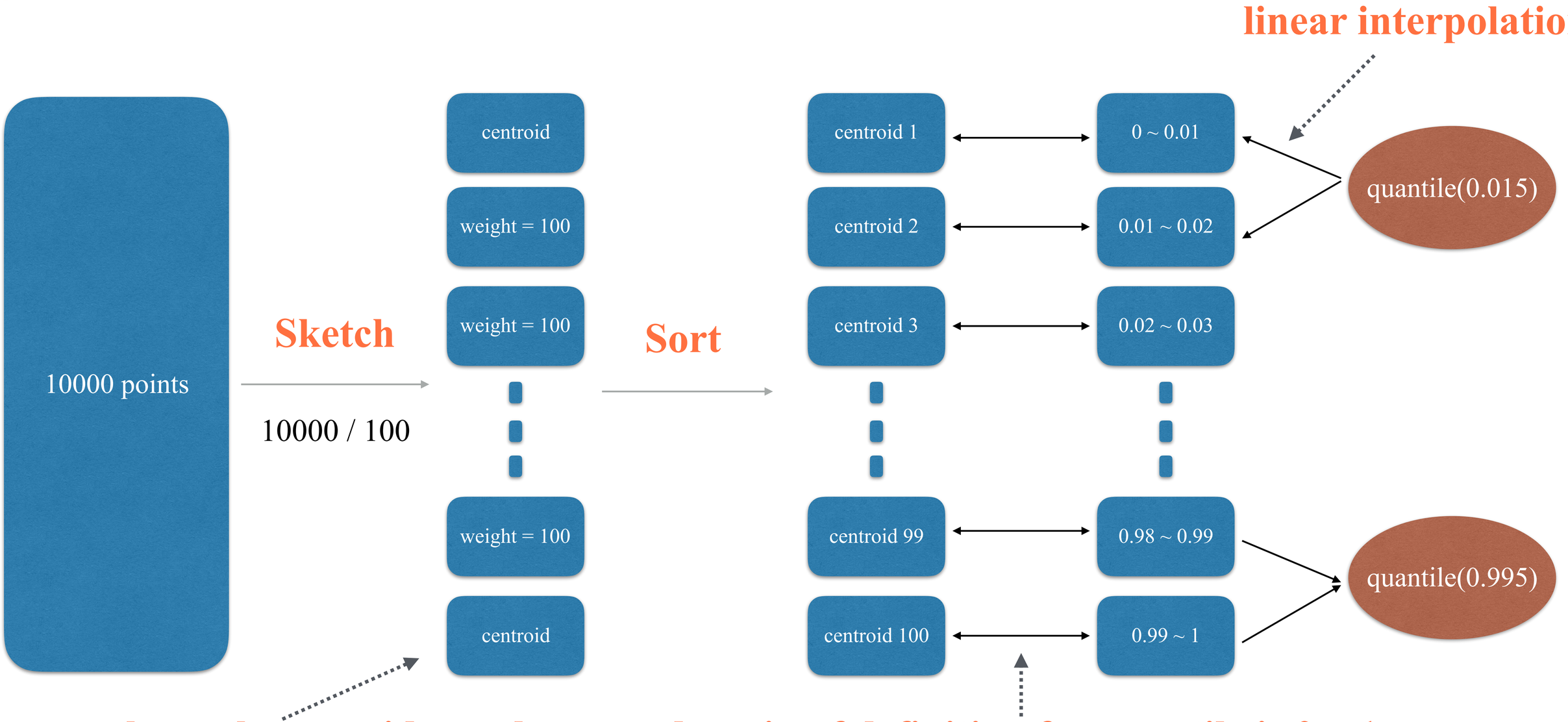


Too small clusters



The Key is how to control cluster size

How to compute quantile from centroids

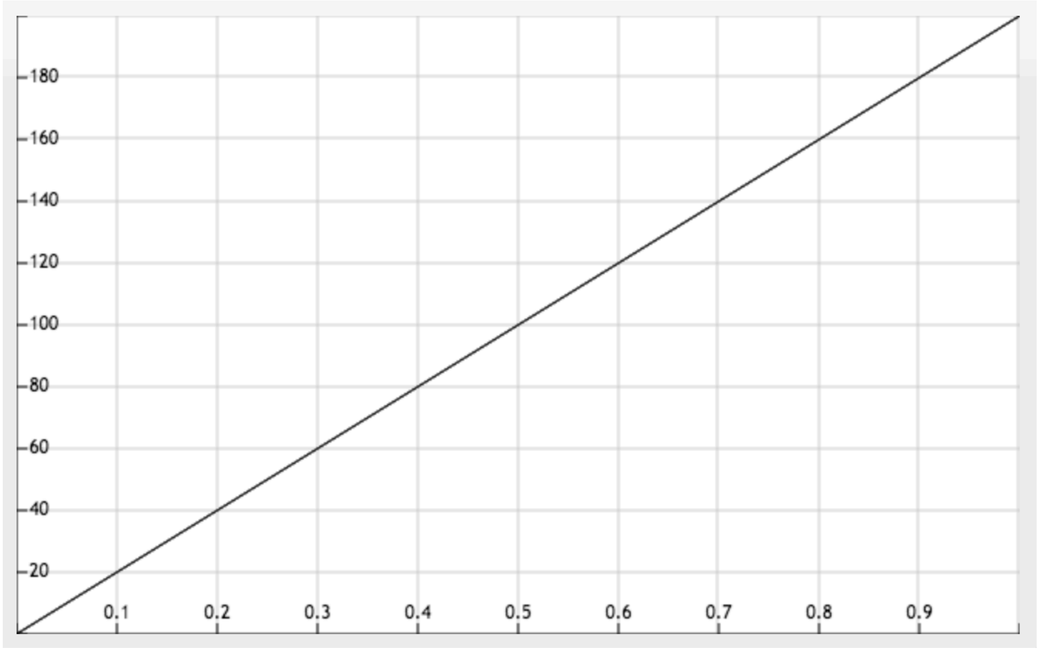


Two concepts

1. **compression:** control the centroids number (control the accuracy)
2. **K(q, compression):** a mapping from quantile q to a centroid index k

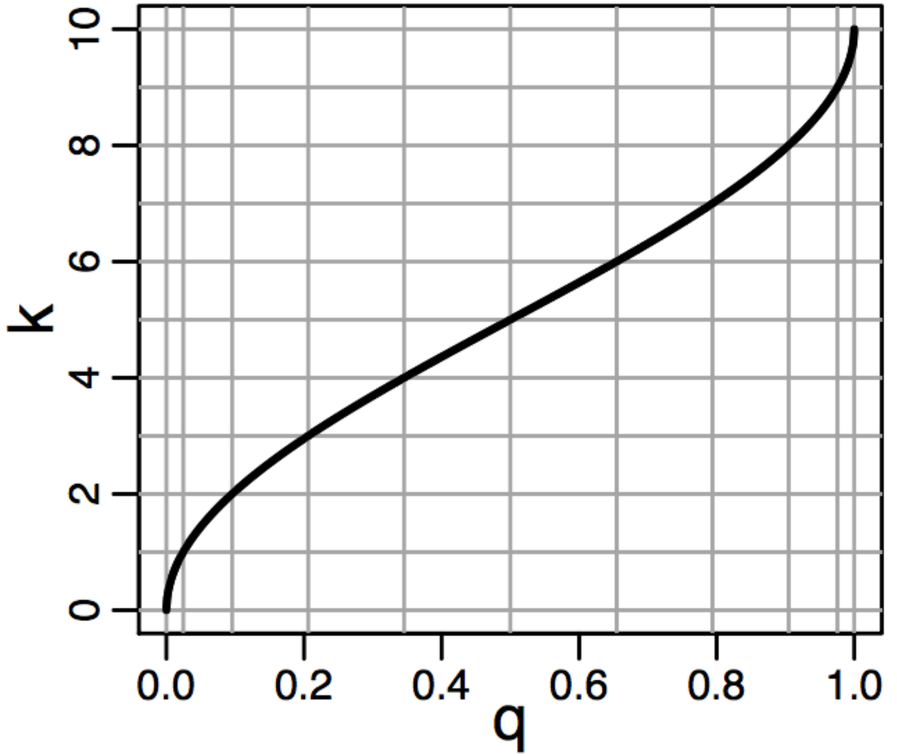
K mapping

We use in example



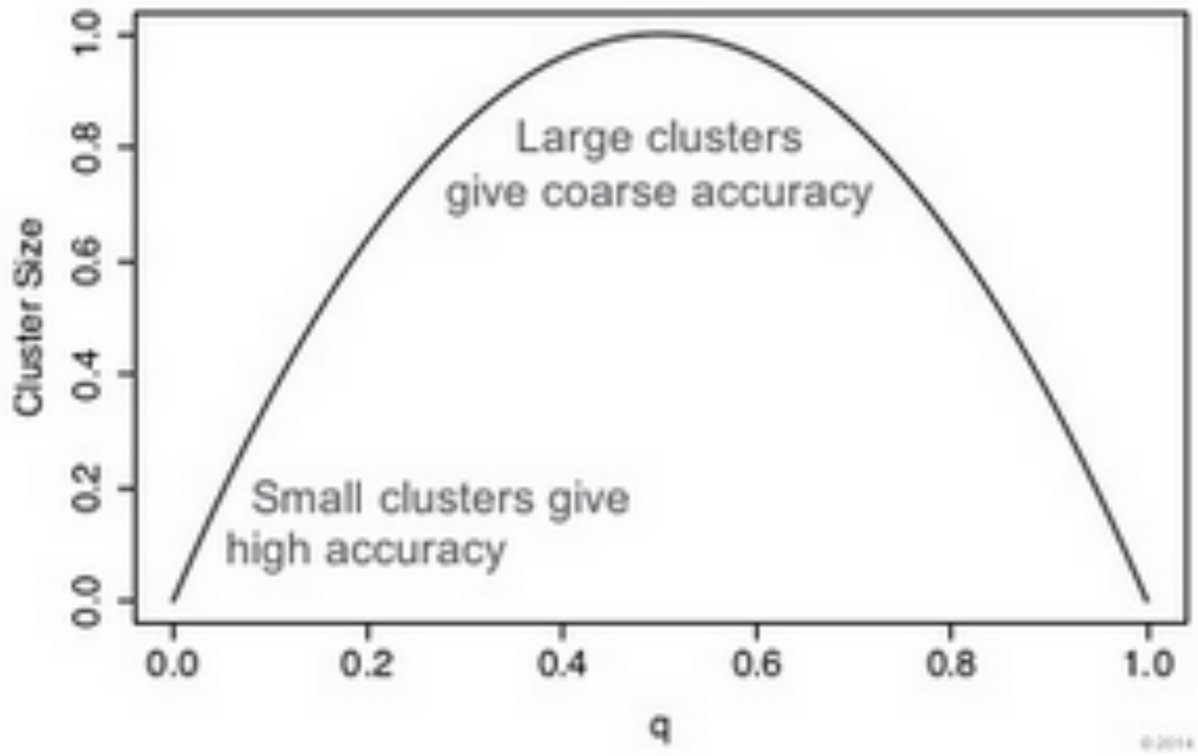
$$k(q,\delta) = \delta * q$$

TDigest **buffer-and-merge** arithmetic use



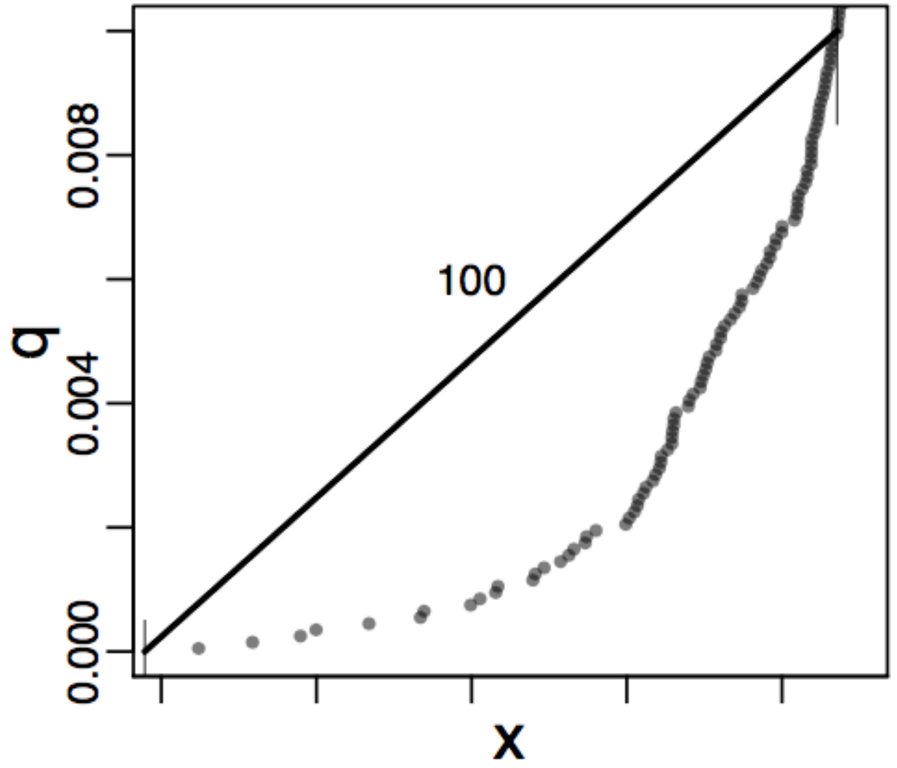
$$k(q,\delta) = \delta \left(\frac{\sin^{-1}(2q-1)}{\pi} + \frac{1}{2} \right)$$

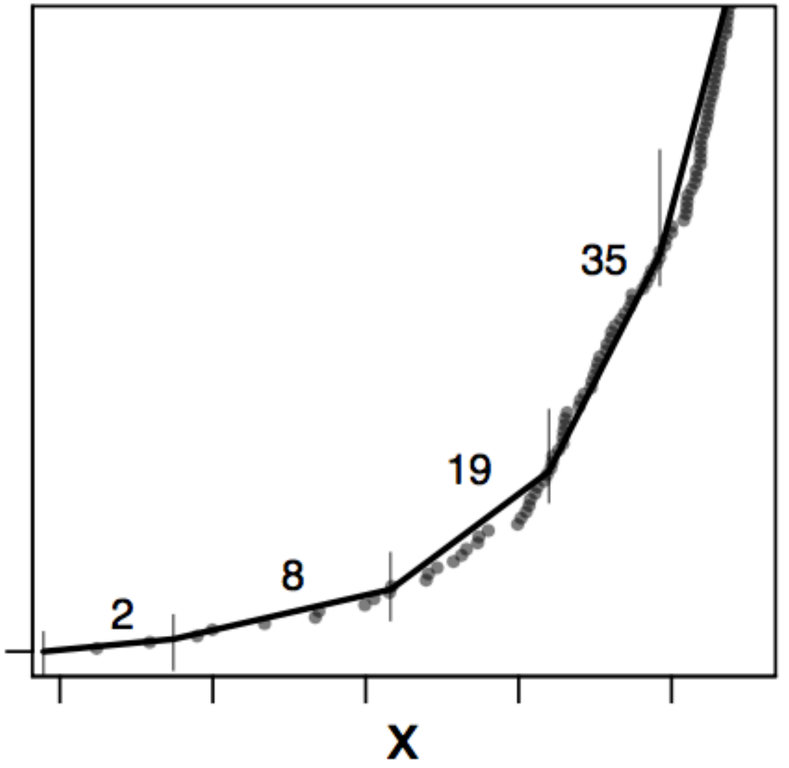
The advantage of buffer-and-merge k mapping



The accuracy near q = 0 or q = 1 is very, very fine

The advantage of buffer-and-merge k mapping

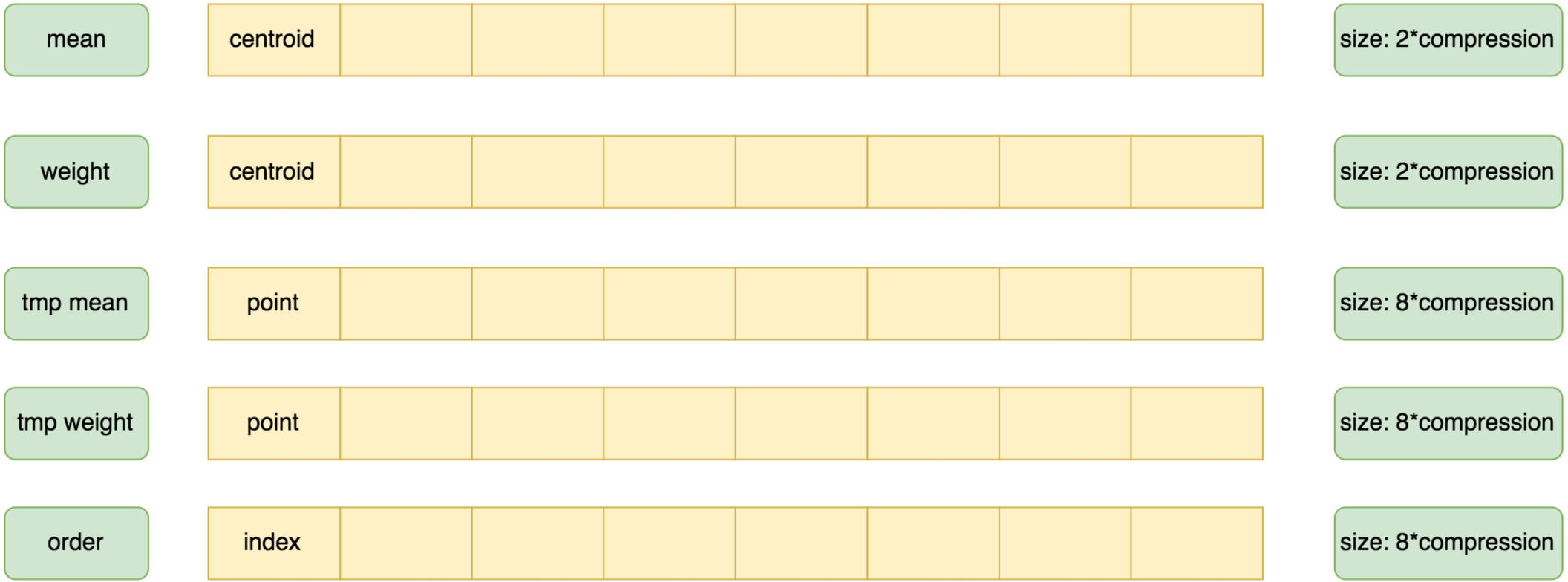




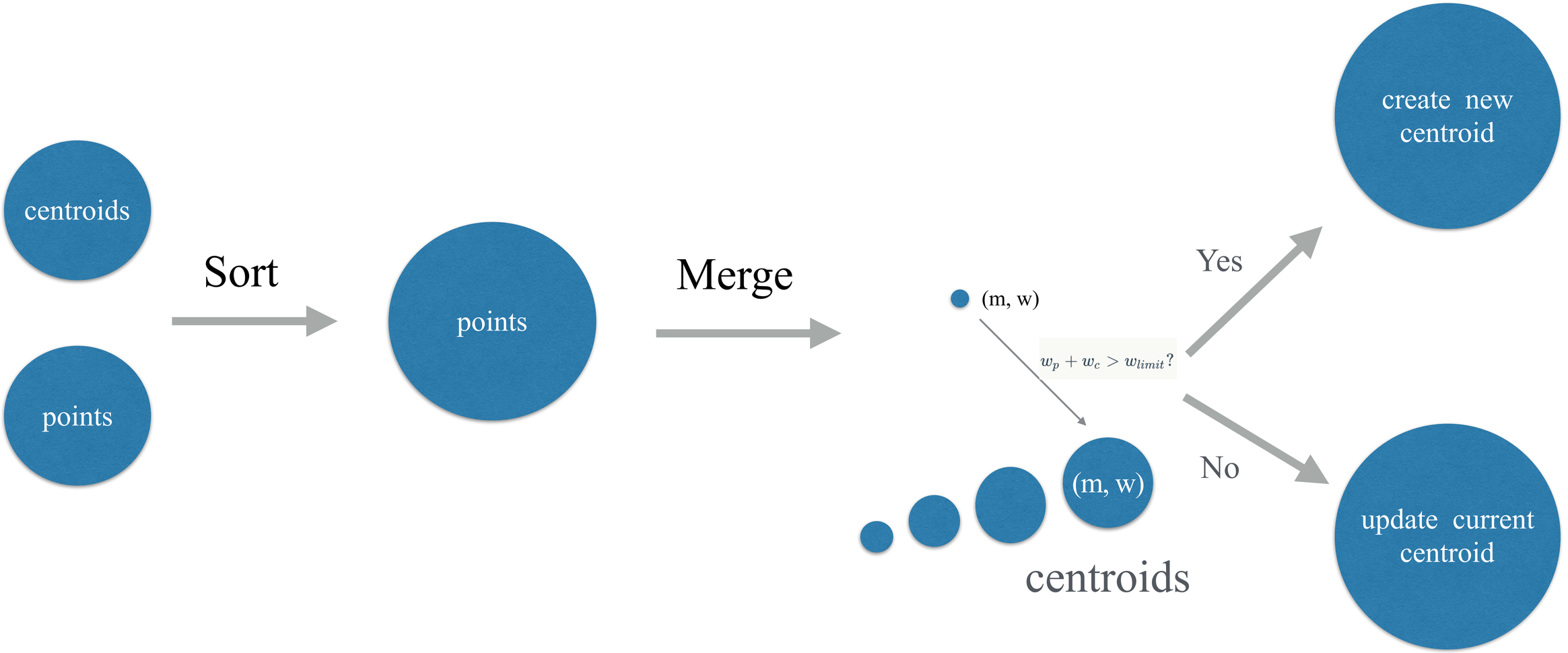
$$k(q,\delta) = \delta * q$$

$$k(q,\delta) = \delta\left(\frac{\sin^{-1}(2q-1)}{\pi} + \frac{1}{2}\right)$$

The data structure of buffer-and-merge

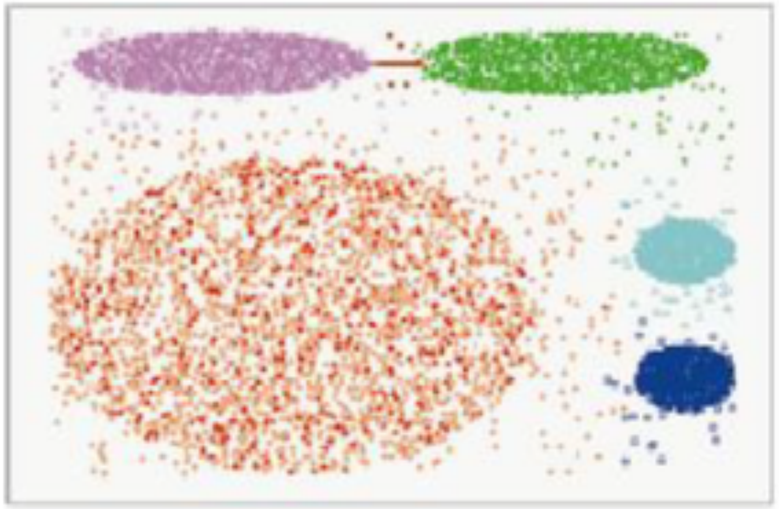


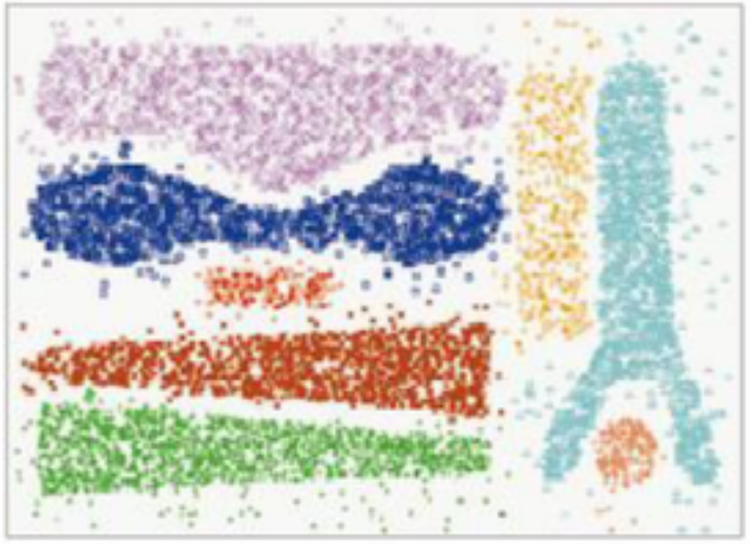
The core of buffer-and-merge



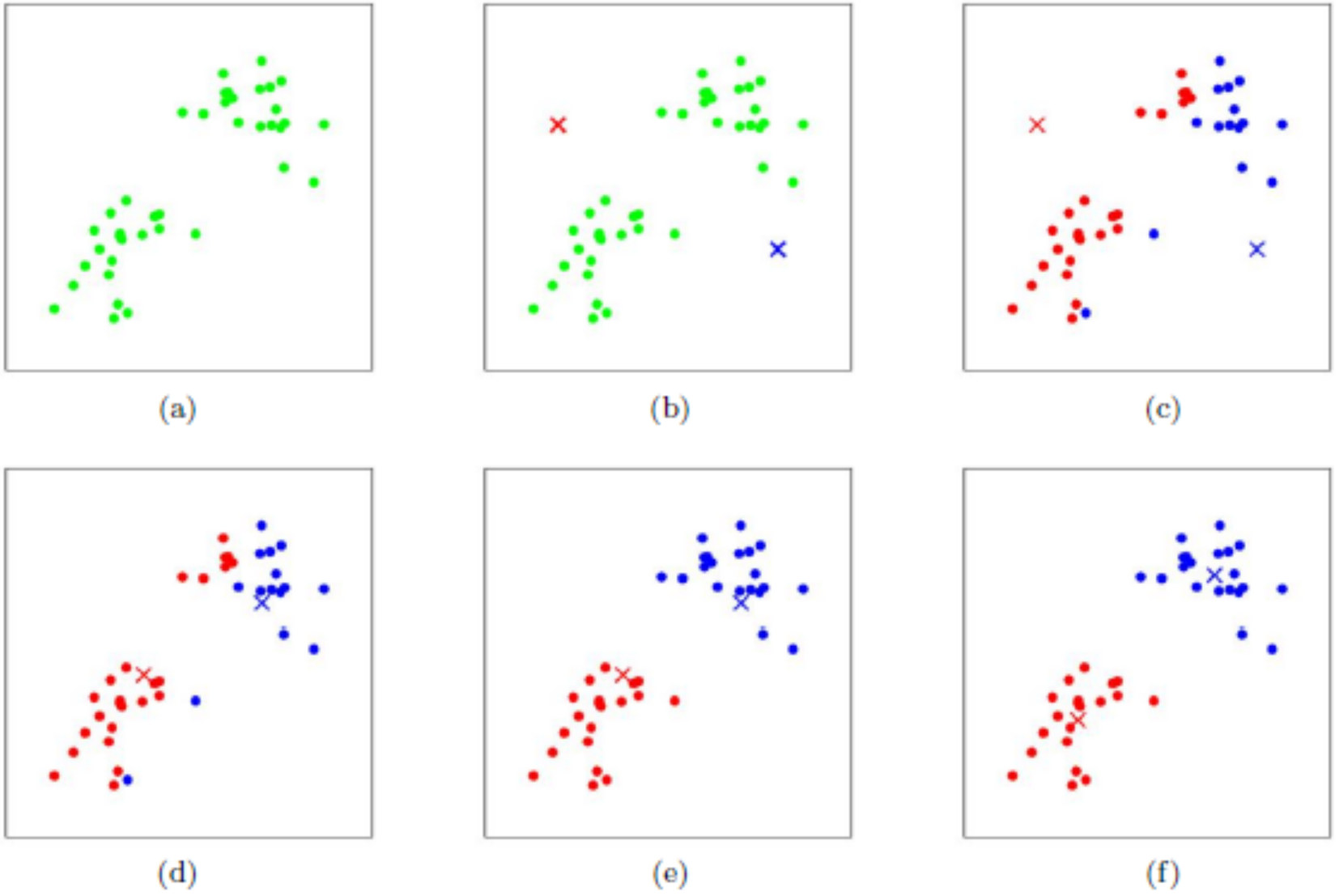
TDigest

Clustering

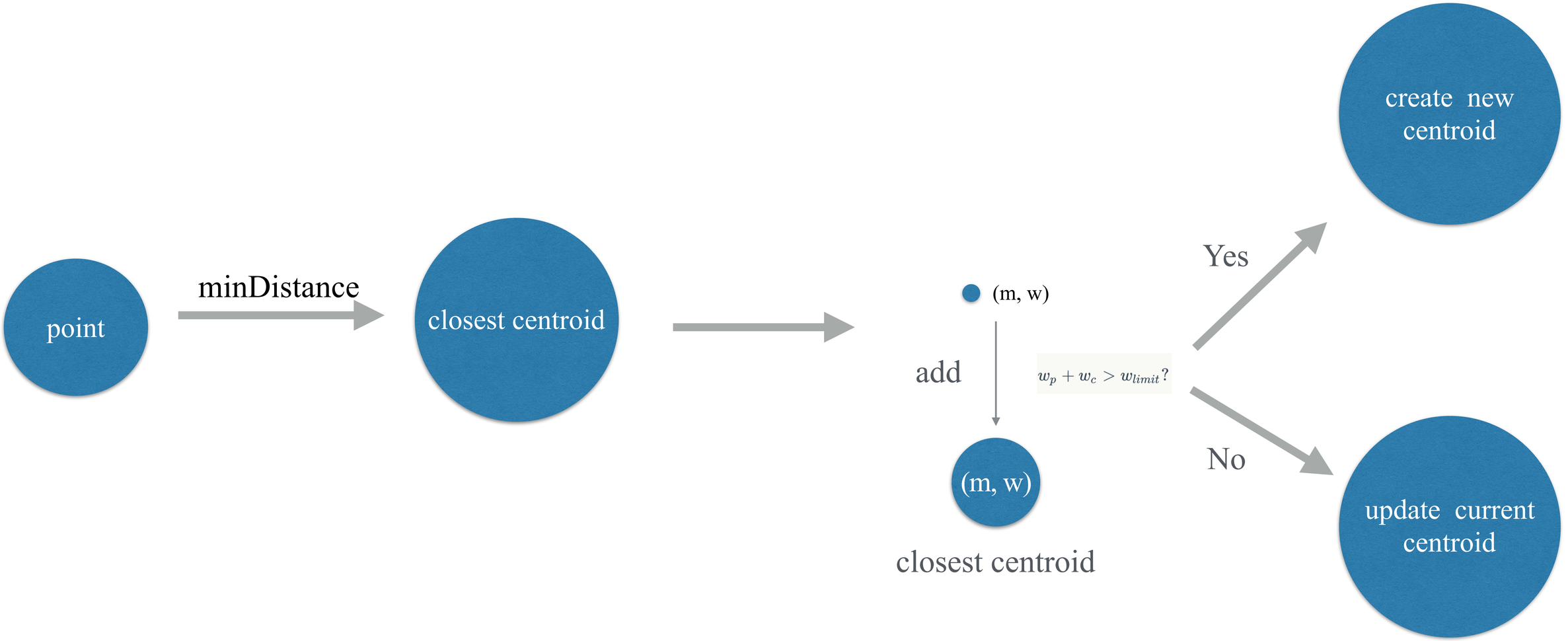




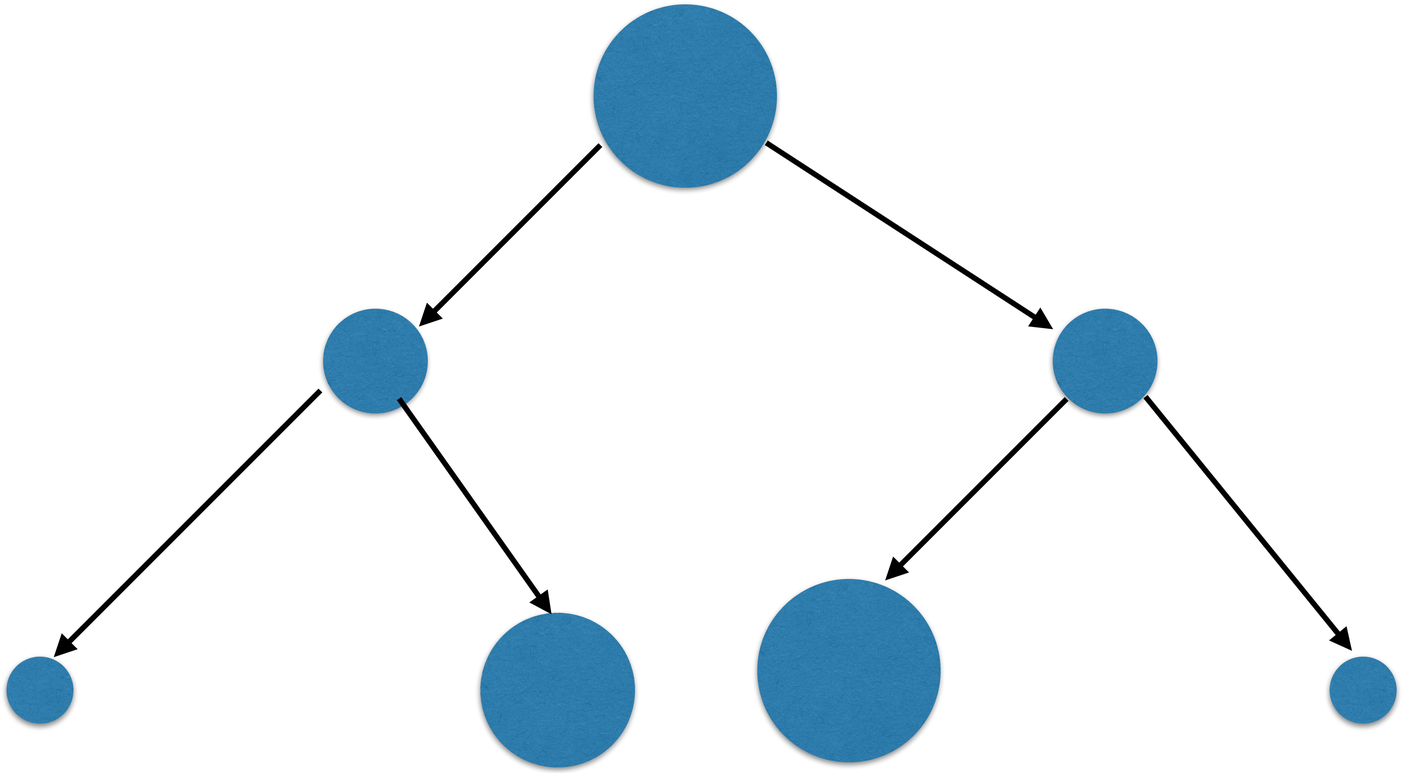
KMeans



The core of clustering



The data structure of clustering: AVL Tree



TDigest

TDigest

- Accurate
- Fast
- Simple
- Monoidal (Map-Reduce)

Conclusion

- Probability and Statistics is great and interesting
- The more big-data and real-time, the more we need sketch

Reference

- Tdigest paper
- Tdigest code
- Some-important-streaming-algorithms-you-should-know-about
- Wikipedia

Thanks

Eat better,Live better.

美团点评

Q&A