面向大数据的K-means算法综述

任远航

(电子科技大学信息与软件工程学院 成都 610054)

摘　要：随着信息技术的发展，移动互联网和物联网等领域正经历着迅猛的发展，海量的数据伴随产生，在海量数据下挖掘数据的模式正变得日益迫切与重要。聚类作为一种重要的数据挖掘方式，其在过去的几十年间一直被广泛研究。在所有的聚类问题中，K-means问题可能是最知名的一个。如何在海量数据下更快获得一个有理论保证的K-means的近似解则是一个关键问题，本篇文章将按照以下方式对这一问题的进展进行综述。首先，文章将定义K-means问题并介绍相关背景；然后，从理论保证和加速两个方面分别介绍国内外先进研究成果；最后，总结现有的成果并对未来的方向予以展望和预测。

关键词：聚类；K-means；采样；次线性时间算法；理论保证

中图分类号：TP181

A survey of K-means algorithms on big data

Yuanhang Ren

(School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China)

Abstract: With the rapid development of the information technology, areas like Mobile Internet and Internet of Things are growing fast. Driven by these areas, more and more data has been generated and large amounts of data need to be processed. Hence, data mining is becoming more and more important under this big data scenario. As a crucial data mining technique, clustering has been studied over the past few decades. Among all the clustering problems, the K-means problem is probably the most well-known one. How to obtain a theoretically guaranteed solution of K-means efficiently for the big data can be a key problem and the progress of this problem is surveyed in this paper with the following steps. First, the paper will define the K-means problem and introduce relevant backgrounds. Second, techniques for theoretical guarantee and speed up will be introduced separately and described in details. Finally, the main results will be summarized and the future directions will be forecasted.

Key words: clustering; K-means; sampling; sub-linear time algorithms; theoretical guarantee

1. 引言

在很多研究领域，研究进展的取得往往需要依赖大量数据的分析。比如在天体物理学领域，为了给黑洞“照相”，Bouman等人需要处理5PB的数据（<http://zhidx.com/p/144996.html>）。　在生物医学领域，一个人体样本的基因组数据会超过100GB，由于一次实验会收集成百上千的人的数据，因此数据量十分巨大 [1]。此外，由于记录数据的设备越来越多其他领域数据量也在不断扩大，比如传感器记录的行动数据、基因数据、照片、语音、金融日志、网络数据等。

海量数据被冠以术语Big Data，Big Data至少包含３层含义 [2]，即数据量大（Volume of data）、数据处理速度快（Velocity of processing the data）与数据多样（Variety of data），合称“3V”。对于这样的数据往往需要使用一些自动化的方法来分析数据中重要的模式和子结构或者对数据进行压缩，这些方法包括聚类和降维，前者即是将数据分为不同的类，使得同类的数据相似，不同类的数据不相似，后者是将数据投影到一个低维空间使得高维空间的数据的结构能够在低维尽可能保留。对于聚类经典的方法有K-means [3]、Spectral Clustering [4]等，经典降维方法有PCA [5]、ISOMAP [6]、LLE [7]等。

本篇文章聚焦聚类问题[8~10]，K-means问题作为经典的聚类问题其解法被广为研究[11~13]。然而，文献[14~16]说明该问题是NP难的，所以诞生了许多近似算法，其中最有名的算法可能就是Lloyd算法 [17]，它分为以下三步。首先，均匀不放回选择k个点作为初始中心点，接着，所有点靠到离它最近的点上形成k个类，最后，计算k个类的中心点作为新的k个点。算法的2、3步一般会重复t次。然而，这个算法有两个缺点。

首先该算法时间复杂度为，其中n是数据量大小，d是数据维度。该算法在海量数据下会花费很多时间，因此更快的算法需要被提出。更严重的是，文献 [18]指出这个算法得到的解的质量可能很差，而在真实的海量数据下解的质量是非常重要的。因此，本篇综述将系统**介绍能够对K-means问题解的质量给出保证的并且尽可能快速的算法**。首先文章将定义K-means问题然后从质量保证和加速两个方面分别介绍算法，最终给出该方向上的展望和总结。

1. K-means问题引入及背景知识

这里首先引入K-means问题。给定点和集合，可以定义点到集合的距离



给定n个点和集合，可以定义如下目标函数



K-means问题便是在给定n个点X情况下找到大小为k的集合使得该目标函数最小，即



上文谈到解的质量，这里给出如下定义用于定量评价解的质量。

**定义1 K-means问题解的质量** 给定数据集X，如果存在一个数字使得成立，则称解C为一个近似解，产生该解的算法称为近似算法，这里称为近似系数。

从该定义可以看出，越小解质量越好，越接近最优解。

1. 给K-means的解提供质量保证
   1. K-means++

选择好的初始化方法（seeding）是提供质量保证的一条路线，在Lloyd算法中传统初始化方法一般是均匀不放回的抓取k个点，这不是一个好的初始化方法，由于Lloyd算法对初始点很敏感 [19]，这种方法为了取得好的效果经常需要多次初始化取最好的，除去花费不少时间此种方法没有理论上的保证，而David等人 [20]则在这一方向上做出了巨大的贡献。

首先，从直觉上想，当初始化的点聚在一起的时候该初始化就不是一个好的初始化，因此一种容易相到的就是每次新选择的点都应离已经选择的点尽可能远，但是如果直接选择最远的点的话就可能是异常点，为此可以这样改进该算法，每一个点以一个概率被选中，而该概率与该点到已选择到的点的距离成正比，由此便得到了K-means++算法 [20]，算法详细描述如算法1

算法1 K-means++ seeding

输入: 数据集，类数目

输出: 解集

a) 均匀从中采样一个点

b) 

c) for = 2,3,…, do

d) for  do

e) 

f) 以的概率从中采样一个点

g) 

该算法的时间复杂度为，实验验证在使用K-means++ seeding后Lloyd算法能更快收敛。另外，对于K-means++以下定理成立。

**定理1** **K-means++解的质量** 对于任意数据集，有以下结论



其中是算法1返回的结果

可以看出即使以初始化的k个点作为最后的解，解的平均质量已经能够达到，后续Lloyd算法的步骤只会让解更好。该方法虽然有好的理论保证，但是由于算法序列化的特征实际花费的时间可能很多，文献[45]后来对此做了改进提出了K-M算法。

* 1. 一个启发式的本地搜索算法

现在，能否获得更好的常数近似的解呢？设想现在已经有一个解的集合，为了在基础上得到更好的解，可以采取以下策略，将点替换为得到，如果的解质量更好，则用替换，重复该步骤直到解质量不再变化。算法描述见算法2

算法2 启发式本地搜索算法(p-swap)

输入: 数据集，类数目

输出: 解集

a) 从中随机选个点

b) While，使得

c) 

在实际应用中这个替换会持续很久，不过，通过对文献 [21]和文献 [22]的结果进行拓展，该算法可以在多项式时间（polynomial time）内收敛。首先，对于任意1次替换不能再减少K-means目标函数值的解称其为1-stable的解。基于此，Arya等人 [21]和Kanungo等人 [23]对解的质量给出了如下的理论结果。

**定理2 1-stable的解的质量** 令，且是所有大小为的的子集里能取得最小K-means目标函数值的那个解，令是1-stable的解，则有



值得注意的是并不是，通过文献 [24]简单的推论，得知近似系数是50。那对于任意p-stable的解，近似系数是多少呢？Kanungo等人 [23]对算法2做了推广和分析得到以下结论。

**定理3 p-stable的解的质量** 令，且是所有大小为的的子集里能取得最小K-means目标函数值的那个解，令是p-stable的解，则有



可见对于充分大的p，系数是18。实验表明算法2收敛相对于Lloyd算法收敛慢，但是该算法能够避免陷入局部极小值。所以，实际使用时会和Lloyd算法混合使用，即先用一次替换再进行Lloyd算法的2、3步，这样的混合算法在实际中有很好的表现，文献 [23]对此做了验证。该算法优点是有更好的解的质量，缺点是相对K-means++更慢。

* 1. K-means++和本地搜索的结合

K-means++虽然实用，但是近似系数只有，本地搜索算法虽然有常数的近似系数，但是不实用，时间复杂度有，能否将这两种算法结合起来得到一个既有常数近似又快速实用的算法呢？Silvio等人 [25]便带来了一个这样的算法，算法描述见算法3。

算法3 K-means++和本地搜索的混合算法

输入: 数据集，类数目，循环数

输出: 解集

a) 将应用于K-means++ seeding算法

b) for  = 2,3,… do

c) = LocalSearch++()

算法4 LocalSearch++算法

输入: 数据集，解集

输出: 解集

a) 以的概率采样

b) if  s.t. 

c) 选择一个使得最小

d) 

**定理4 K-means++和本地搜索的混合算法的解的质量** 令是算法3的解，且，则有



算法3的理论保证由定理4给出，定理4说明只要迭代大概次LocalSearch++，解的近似系数就能达到常数，此时算法3的时间复杂度是。实验表明在不使用Lloyd算法后续迭代的情况下，混合的算法3比K-means++好8-35%，而使用Lloyd迭代10次后，依然能比K-means++好1-18%，该算法在常数近似的情况下大幅减少了本地搜索的时间，又不比K-means++慢很多，实际情况下推荐使用。

* 1. 实验效果

本小节对上述算法进行实验对比分析，本节用到的数据有intrusion数据集，KDD-PHY数据集和RNA数据集。intrusion有494019个点，每个点有35维，KDD-PHY有100000个点，每个点有78维，RNA有8个特征，488565个点。首先，将K-means++和K-means在intrusion上比较得出以下表格

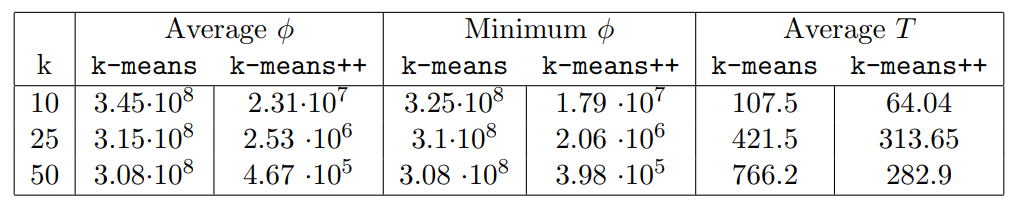


表1 K-means++和K-means实验比较

Tab. 1 Compare K-means++ with K-means on intrusion datasets

K-means++结果令人惊叹，可以看到K-means++距离平方和比K-means小10到1000倍，并且快了有70%，接着在KDD-PHY和RNA上比较K-means++和K-means++&swap。

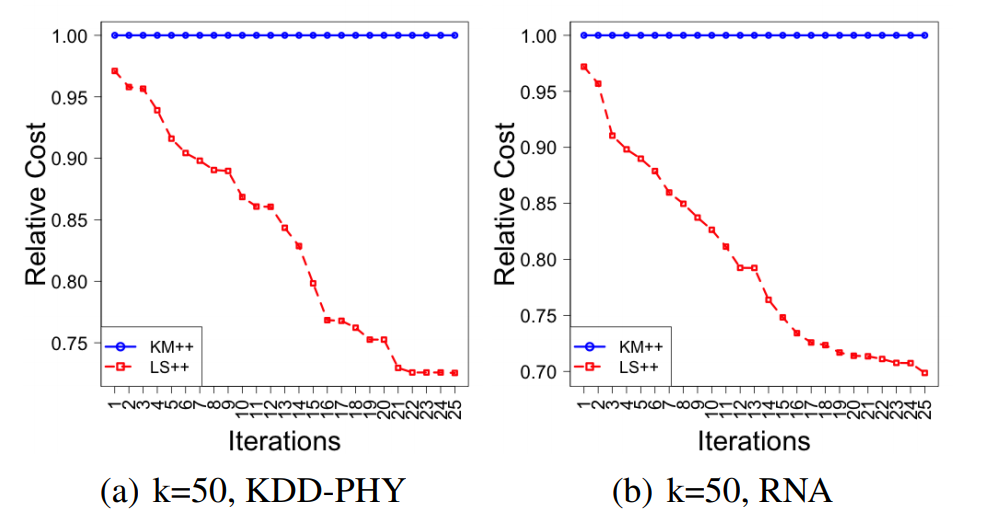


图1 K-means++和K-means++&swap实验比较

Fig. 1 Compare K-means++ with K-means++&swap on KDD-PHY and RNA datasets

可以看到随着循环数的增大，LS++（K-means++&swap）的距离平方和近乎线性减少，显著优于K-means++。

* 1. 小结

至此，本节梳理了一些有理论保证的算法同时在实验上对他们进行了比较，值得注意的是由于K-means问题是APX难的（文献 [26]和 [27]），多项式时间内不存在近似系数小于1.0013的解，目前文献 [28]给出了最好的近似系数6.357。不过，近似系数越小的算法越慢，越难以实用。另外，K-means问题可以构造PTAS的解，即在多项式时间内取得的近似，相关文献有[29~31]。在理论保证的方向上，未来趋势将会是在常数近似的情况下，提出更简单实用，易于并行的算法。

1. 更快得到K-means的解

本小节将从减少数据量和给数据做假设两方面来介绍如何更快得到K-means问题的解。

* 1. 减少数据量

加速聚类的一个自然的想法是将大数据浓缩，得到小数据，在小数据上聚类得到的解作为最终解。这里介绍三种方法。

* + 1. 均匀采样

一种简单且有理论保证的采样方式是均匀采样 [32]，即首先在所有点中均匀采样若干点，然后在这些点上聚类，算法详细描述见算法5

算法5 基于均匀不放回采样的聚类算法

输入: 数据集，采样数，类数目

输出: 解集

a) 从中均匀不放回采样个点

b) 在上运行一个近似算法

可以证明该方法有如下的理论保证

**定理5** **均匀不放回采样的解的质量**令是近似的参数。令是由算法5返回的解. 假设算法均匀不放回的采样个点，使得



则有



以至少的概率成立，其中是数据直径的平方，是点到其对应的中心点的距离的平均值。

该方法优点是简单，可以看到如果在采样点上使用有理论保证的算法就可以使得得到的解对所有点也是有保证的，但是该方法的采样点由数据点的直径决定，如果直径较大，该方法便会失去实用价值。类似的还有基于均匀放回采样的算法 [33]

* + 1. K-means||采样

由于在均匀采样里存在直径问题，所以需要提出新的理论结果不依赖直径的算法。前述K-means++体现了基于距离采样的优势，由此基于距离采样的K-means||算法便被提了出来 [34]，它的想法是基于已经挑选出的点集，下一次不止挑选一个点而是挑选一个集合，其期望大小为，将若干次挑选出来的集合汇总之后再在其上做聚类即可，采样算法详细描述见算法6，对于算法6有如下理论保证

算法6 K-means||采样

输入: 数据集，每一轮采样数

输出: 采样集

a) 在中均匀采样一个点

b) 

c) for  = 1,2,…do

d) 

e) for  do

f) 以概率将加入中

g) 

**定理6 K-means||采样的理论保证** 令是算法6 次迭代后返回的结果，则有



其中

上述定理说明算法6经过次迭代之后，所有点到挑选出来的S个点的距离和已经降到了，也就是说和最优的距离和已经在一个级别了。接着在S上聚类便可以得到K-means的解，在这里为了使解有理论上的保证，需要对S上的点加权重并定义如下带权K-means问题。

**带权K-means问题** 给定集合和每一个的权重 , 找到一个大小为k的集合使得下述目标函数能最小



接着在带权重的S上聚类得到算法7

算法7 基于K-means||采样的聚类方法

输入: 数据集，每一轮采样数

输出: 解集

a) 将应用于算法3

b) 对点, 令是中离最近的点的数目

c) 令是的权重，在带权的上运行一个近似算法

可以证明，如果使用的是近似算法，那算法7输出的k个点便是原K-means问题的一个近似解。在实践中发现当 较大时，往往不需要次迭代，O. Bachem [35]的分析证实了这一点，令迭代次数为，则有下述定理。

**定理7** **K-means||采样的新理论保证** 令且，令且是算法6返回的结果，则



该定理同定理6关键区别是只要够大，指数衰减的速度可以任意快，而定理6的衰减速度最多是1/2。虽然K-means||时间复杂度是（如果次迭代则是），但是它易于并行，因此在能够并行的情况下很多经典大数据处理框架都采用了该算法比如Spark[36,37]。同均匀采样一样，如果在采样点上使用有理论保证的算法就可以使得到的解对所有点也是有保证的。该方法的优点就是易于并行，缺点是相较于前述均匀采样算法更耗费运算资源。

* + 1. Coreset

前述算法得到的采样点都有一个好的性质，即在采样点上运行的算法得到的解对全部点是有理论保证的，这些采样点都属于Coreset的特例。通俗来讲，Coreset指的是一些特殊的采样点（可能带权重），它有一个强大的性质是，如果在这些采样点上运行一个近似的聚类算法，则能获得一个在全部点上是近似的解。比如目前效果很好的-Coreset，它基于重要性采样(importance sampling)。算法见文献 [38]中的算法9。其采样数为，时间复杂度为，对于采样点来说近似系数是。该方法的近似系数非常好而花费的时间又是线性的，可以说是目前K-means问题中相当好的算法。

虽然常规的Coreset如-Coreset虽然效果好，但是花费的时间还是稍微有点多，Lightweight Coreset [39]即是牺牲了一部分聚类的质量来换取速度的提升，具体来说Lightweight Coreset指的是满足如下定义的带权点集。

**定义2 Lightweight Coreset 令**，，是中的数据点，它中心点是。如果对任意有



则带权点集称为一个-Lightweight Coreset。

由于-Lightweight Coreset对于任意个点的集合 都有以上属性，因此易得其有以下近似系数的推论。

**推论2.1** 如果是一个-Lightweight Coreset，有



其中是上的K-means问题最优解，是上带权K-means问题最优解。

算法8 Lightweight Coreset

输入: 数据集，采样数

输出: 采样集

a) 的均值

b) for  do

c) 

d) 以的概率从中采样个点，每个点权重是

构造该Coreset的算法详述见8，关于该算法有如下结论。

**定理8** **Lightweight Coreset的理论保证** 令，且，是中的数据点，是算法8的输出。其中采样数



是一常数，那么点集将会以至少的概率成为一个-Lightweight Coreset。

该定理说明，算法只要在的时间复杂度里采样常数个点（与n无关），在采样点上聚类就能取得好的解（如果不大的话）。

构造Coreset的方法除了这里提到的，还有很多，比如还可以用K-means++构造Coreset [40]，Coreset作为一个经典的数据压缩范式还被用于其他机器学习问题中，比如PCA，NNMF（Non Negative Matrix Factorization）等，文献 [41]和 [42]是很好的参考资料。构造Coreset的算法基本是目前采样中效果最好的，缺点就是不易理解算法原理。

* 1. 对数据做假设

可以看到不管是上面三种方法中的哪一种，都没有办法将时间复杂度缩小至线性时间以内，那么存在次线性时间（sub-linear time）的方法吗？遗憾的是，Mettu等人 [43]说明，如果想获得常数近似的解，在没有对数据做假设的情况下，任意算法都需要的时间复杂度。这说明，对数据做假设是一种更快的获得解的方法。

通过分析即可知道在K-means++方法中最费时间的是找到计算概率所需要的分母，有没有在不知道分母的情况下近似采样分布的方法呢？有，该算法便是MH算法 [44]，该算法属于MCMC方法，其基本想法是构建一个马尔可夫链使得其平稳分布（stationary distribution）等于要采样的分布，这样就可以通过随机游走的方式来近似采样，基于MCMC来近似K-means++的方法称为K-M [45]，Bachem等人通过对数据集生成的分布做出假设使得基于马尔科夫链随机游走的时间复杂度被限制在次线性时间内，从而让K-M在的时间复杂度下获得了K-means++的近似系数。K-M算法见算法9。

算法9 K-Mseeding

输入: 数据集，游走次数

输出: 解集

a) 在中均匀采样1个点

b) 

c) for = 2,3,…,k do

d) 在中均匀采样1个点

e) 

f) for = 2,3,…,m do

g) 在中均匀采样1个点

h) 

i) if  then

j) 

k) 

这里定义两个数据相关的指标



假定数据是从生成数据的分布中独立同分布的采样出来的，且该分布满足以下两个假设，如果满足这些假设的话和就可以被限定住，游走就能很快收敛到平稳分布

**假设1** 分布的尾部是指数衰减的，即使得，，其中

**假设2** 在一个球面上的最大和最小密度被一个常数限制

如果满足假设1，可以证明，如果满足假设2，可以证明，这样就有如下定理。

**定理9** **K-Mseeding的理论结果** 在假设1和2成立的情况下，



为算法9的返回结果，此时，算法时间复杂度为。该方法在次线性时间里完成了的近似，因此在数据较大且不能很好并行的情况下应该优先使用该算法。然而，在实践中该方法有以下缺点

1. 如果存在小的类且该类离其他类比较远，那由于均匀选择的点很难落在小的类里，这样该方法就会产生较大的距离和
2. 不少现实中的数据不满足上述假设，比如经常能观测到重尾分布（heavy tailed distributions）的数据
3. 验证上述假设是困难的，比如计算需要对数据进行两次遍历，计算更是NP难的
4. 定理9并没有描述游走次数和算法解的质量之间的权衡，只有选择特定的游走次数才有效，因此，如果假设不成立，就得不到任何理论上的保证

因此，需要提出新的算法解决这些问题。

AFK-M [46]通过使用提议分布（proposal distribution）很好的解决了这些问题，首先通过该提议分布，的假设得以废弃，第二，一种新的理论分析使得在不假设的情况下也能获得解的质量的理论保证，最后，新的结果表明了游走次数和算法质量之间的权衡。AFK-M描述如下

算法10 AFK-M seeding

输入: 数据集，游走次数

输出: 解集

*// 预处理*

a) 在中均匀采样1个点

b) for  do

c) 

*// 主循环*

d) 

e) for  = 2,3,…,k do

f) 以的概率从中采样一个点

g) 

h) for  = 2,3,…,m do

i) 以的概率从中采样一个点

j) 

k) if  then

l) 

m) 

可以看出点的选择依赖提议分布，因此小的类也能被选到点，从而解决了K-Mseeding方法中的问题1。对于AFK-M，有如下定理

**定理10 AFK-M的理论结果** 令且，数据集，， 是算法10的输出，，则



其中

容易分析得，算法中预处理花费时间为，主要循环花费时间为。虽然预处理需要花费线性的时间，时间复杂度比K-M高，但实际情况中有以下理由说明该步骤问题不会很大

1. 对所有数据只需遍历一次
2. 该步骤很容易并行
3. 从定理可以看到，预处理后游走的步数从变到了常数

因此该算法和K-M一样，在实际中有良好的适用性，另外，定理10良好的揭示了游走次数同聚类质量间的联系，可以看到随着的减小，游走步数逐渐增大，聚类质量逐渐向K-means++的质量靠拢。同时，如果对数据的假设满足的话，可以得到下述推论。

**推论10.1** 令，数据集，，且满足，是算法10的输出，，则



此时，算法中预处理花费时间为，主要循环花费时间为。实验结果表明，对于大数据集，在0到1%的相对误差下，AFK-M算法要比K-means++快200到1000倍。

除去前述的假设1和2，也有些其他假设值得注意。Ben-David [47]通过引入聚类描述方案（clustering description scheme）并在其上添加性质，获得了常数时间复杂度的算法，且该算法有常数近似系数的解。文献 [47]的主要假设叫做-m-covering，它要求对领域集合（domain set）中的任何子集，通过应用聚类描述方案到的一个元组上，的最优聚类能被近似。

* 1. 实验效果

本小节对加速算法的实验结果进行比较分析，使用的数据集有SPAM，KDD，CSN。SPAM有4601个点，每个点58维，KDD有145751个点，每个74维，CSN有80000个点，每个点17维。先对减少数据量的3种算法进行比较。

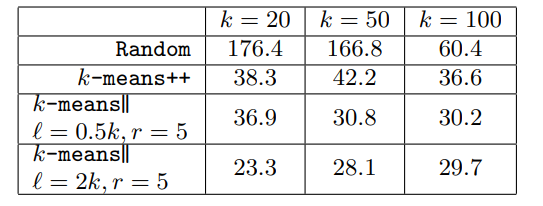


表2 K-means||和K-means++实验比较

Tab. 2 Compare K-means|| with K-means++ on SPAM datasets

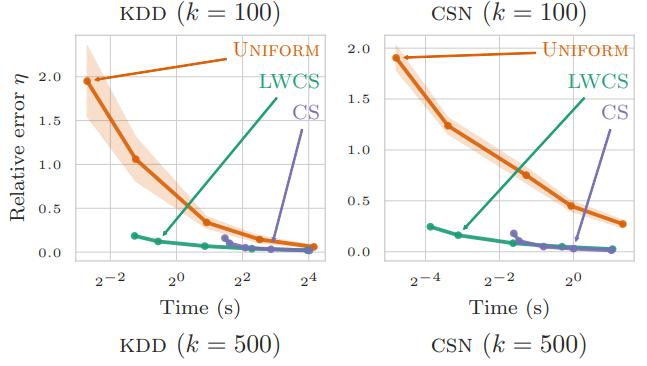


图2 Coreset和Uniform sampling实验比较

Fig. 2 Compare Coreset with Uniform sampling on KDD and CSN datasets

表2展示了K-means||采样在SPAM数据上的效果，对比K-means++，只需更少的迭代次数就能让Lloyd算法收敛，说明其采样更有代表性。图2在KDD和CSN上对比了Coreset和Uniform采样算法，图中的点代表了5种规格的采样数目，分别是{1000, 2000, 5000, 10 000, 20 000}，可以看到，只有当采样数目比较大的时候Uniform采样才和Coreset有相当的聚类效果，Coreset，尤其是Lightweight Coreset能在很少的采样上就取得好的聚类效果。

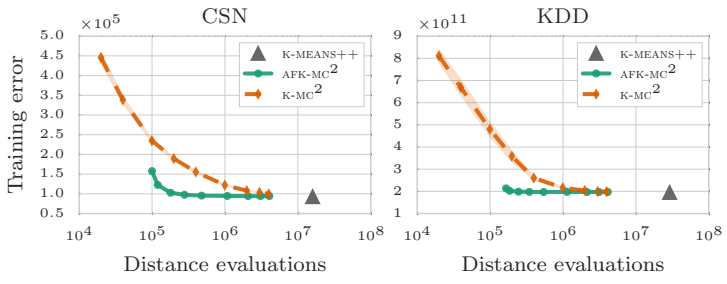


图3 K-M和AFK-M实验比较

Fig. 3 Compare K-Mwith AFK-Mon CSN and KDD datasets

接着比较K-M和AFK-M，结果见图3，横轴代表运行时间，可以看到AFK-M在加了预处理步骤后相比K-M能在更少的时间内取得相同的聚类效果，在能够并行的情况下应该优先使用AFK-M。但是AFK-M依然有缺点，那就是在它的理论保证里依然有方差项，且没有假设无法去除，而该项可能由于数据分布的关系而很大，如何去除该项将会是后续工作之一。

1. 总结及展望

至此本篇文章从有理论保证和加速两个方面梳理了K-means问题的相关算法，相关结论整理在下面的表3中

|  |  |  |  |
| --- | --- | --- | --- |
| 算法 | 采样量 | 时间复杂度 | 近似系数 |
| K-means++ | k |  |  |
| p-swap | k |  | 18 |
| K-means++  &swap | k |  |  |
| Uniform Sampling |  |  |  |
| K-means|| |  |  |  |
| Lightweight Coreset |  |  |  |
| -Coreset |  |  |  |
| K-M | k |  |  |
| AFK-M | k |  |  |

表3 理论保证及加速相关算法结果汇总

Tab. 3 A summary of results on various K-means algorithms

从表3可以看到目前在常数近似的算法里，时间最短且无需假设的是-Coreset，而目前仅有K-M在次线性时间内有理论保证。由于在次线性时间内结果较少，所以在此，文章指出以下可能的研究方向。

1. 在次线性时间内有理论保证且无需假设的算法，比如近似系数是的等等
2. 依靠假设将均匀采样的时间及采样数目限制住
3. 提出新的次线性时间有理论保证的算法，依靠新的易于验证的假设
4. 探索K-means++的新的性质，及其他有用的采样

总的来说，K-means问题的研究趋势将会朝提出更快且有理论保证的算法的方向发展[48,49]，无需或者易于验证的假设以及实用性将会是这些算法的重要考量。随着新的分析技术的发展，K-means问题历久弥新，新的理论结果依然在不断出现，实用的算法层出不穷，随着大数据时代的到来，该问题必将焕发出新的活力。

1. 参考文献
2. 宁康, 陈挺. 生物医学大数据的现状与展望[J]. 科学通报 (中文版), 2015, 60(5/6): 534-546. (Ning Kang, Chen Ting Big data for biomedical research: Current status and prospective[J] Chinese Journal 2015, 60(5/6): 534-546)
3. Beyer M A, Laney D. The importance of ‘big data’: a definition[J]. Stamford, CT: Gartner, 2012: 2014-2018.
4. MacQueen J. Some methods for classification and analysis of multivariate observations[C]//Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. 1967, 1(14): 281-297.
5. Von Luxburg U. A tutorial on spectral clustering[J]. Statistics and computing, 2007, 17(4): 395-416.
6. Pearson K. LIII. On lines and planes of closest fit to systems of points in space[J]. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 1901, 2(11): 559-572.
7. Balasubramanian M, Schwartz E L. The isomap algorithm and topological stability[J]. Science, 2002, 295(5552): 7-7.
8. Roweis S T, Saul L K. Nonlinear dimensionality reduction by locally linear embedding[J]. science, 2000, 290(5500): 2323-2326.
9. 孙吉贵, 刘杰, 赵连宇. 聚类算法研究[J]. 软件学报, 2008, 19(1): 48-61.(Sun Jigui, Liu Jie, Zhao Lianyu, Clustering Algorithms Research[J] Journal of Software 2008, 19(1): 48-61)
10. 贺玲, 吴玲达, 蔡益朝. 数据挖掘中的聚类算法综述[J]. 计算机应用研究, 2007, 24(1): 10-13. (He Ling, Wu Lingda, Cai Yichao, Survey of Clustering Algorithms in Data Mining[J] Application Research of Computers 2007, 24(1): 10-13)
11. 周涛, 陆惠玲. 数据挖掘中聚类算法研究进展[J]. 计算机工程与应用, 2012, 48(12): 100-111. (Zhou Tao, Lu Huiling, Clustering Algorithm Research Advances on Data Mining[J] Computer Engineering and Applications 2012, 48(12): 100-111)
12. 张建萍, 刘希玉. 基于聚类分析的 K-means 算法研究及应用[J]. 計算機應用研究, 2007, 24(5): 166-168. (Zhang Jianping, Liu Xiyu, Application in Cluster's Analysis is Analyzed in Children Development Period[J] Application Research of Computers 2007, 24(5): 166-168)
13. 周爱武, 于亚飞. K-means 聚类算法的研究[J]. 计算机技术与发展, 2011, 21(2): 62-65. (Zhou Aiwu, Yu Yafei, The Research about Clustering Algorithm of K-means[J] Computer Technology and Development 2011, 21(2): 62-65)
14. 吴夙慧, 成颖, 郑彦宁, 等. K-means 算法研究综述[J]. 数据分析与知识发现, 2011, 27(5): 28-35. (Wu Suhui, Cheng Ying, Zheng Yanning, Pan Yuntao, Survey on K-means Algorithm[J] Data Analysis and Knowledge Discovery 2011, 27(5): 28-35)
15. Garey M R, Johnson D, Witsenhausen H. The complexity of the generalized Lloyd-max problem (corresp.)[J]. IEEE Transactions on Information Theory, 1982, 28(2): 255-256.
16. Kleinberg J, Papadimitriou C, Raghavan P. A microeconomic view of data mining[J]. Data mining and knowledge discovery, 1998, 2(4): 311-324.
17. Mahajan M, Nimbhorkar P, Varadarajan K. The planar K-means problem is NP-hard[C]//International Workshop on Algorithms and Computation. Springer, Berlin, Heidelberg, 2009: 274-285.
18. Lloyd S. Least squares quantization in PCM[J]. IEEE transactions on information theory, 1982, 28(2): 129-137.
19. Dasgupta S, “Algorithms for K-means clustering”
20. Cao F, Liang J, Jiang G. An initialization method for the K-means algorithm using neighborhood model[J]. Computers & Mathematics with Applications, 2009, 58(3): 474-483.
21. Arthur D, Vassilvitskii S. K-means++: The advantages of careful seeding[C]//Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, 2007: 1027-1035.
22. Arya V, Garg N, Khandekar R, et al. Local search heuristics for k-median and facility location problems[J]. SIAM Journal on computing, 2004, 33(3): 544-562.
23. Charikar M, Guha S. Improved combinatorial algorithms for the facility location and k-median problems[C]//40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039). IEEE, 1999: 378-388.
24. Kanungo T, Mount D M, Netanyahu N S, et al. A local search approximation algorithm for K-means clustering[J]. Computational Geometry, 2004, 28(2-3): 89-112.
25. Dasgupta S, “The K-means clustering problem”
26. Lattanzi S, Sohler C. A better K-means++ Algorithm via Local Search[C]//International Conference on Machine Learning. 2019: 3662-3671.
27. Awasthi P, Charikar M, Krishnaswamy R, et al. The hardness of approximation of euclidean K-means[J]. arXiv preprint arXiv:1502.03316, 2015.
28. Lee E, Schmidt M, Wright J. Improved and simplified inapproximability for K-means[J]. Information Processing Letters, 2017, 120: 40-43.
29. Ahmadian S, Norouzi-Fard A, Svensson O, et al. Better guarantees for K-means and euclidean k-median by primal-dual algorithms[J]. SIAM Journal on Computing, 2019 (0): FOCS17-97-FOCS17-156.
30. Feldman D, Monemizadeh M, Sohler C. A PTAS for K-means clustering based on weak coresets[C]//Proceedings of the twenty-third annual symposium on Computational geometry. ACM, 2007: 11-18.
31. Jaiswal R, Kumar A, Sen S. A simple D 2-sampling based PTAS for K-means and other clustering problems[J]. Algorithmica, 2014, 70(1): 22-46.
32. Friggstad Z, Rezapour M, Salavatipour M R. Local search yields a PTAS for K-means in doubling metrics[J]. SIAM Journal on Computing, 2019, 48(2): 452-480.
33. Mohan M, Monteleoni C. Beyond the Nyström approximation: Speeding up spectral clustering using uniform sampling and weighted kernel K-means[C]//Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI 2017). 2017.
34. Czumaj A, Sohler C. Sublinear-time approximation for clustering via random sampling[C]//International Colloquium on Automata, Languages, and Programming. Springer, Berlin, Heidelberg, 2004: 396-407.
35. Bahmani B, Moseley B, Vattani A, et al. Scalable K-means++[J]. Proceedings of the VLDB Endowment, 2012, 5(7): 622-633.
36. Bachem O, Lucic M, Krause A. Distributed and provably good seedings for K-means in constant rounds[C]//Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017: 292-300.
37. Zaharia M, Chowdhury M, Franklin M J, et al. Spark: Cluster computing with working sets[J]. HotCloud, 2010, 10(10-10): 95.
38. Zaharia M, Xin R S, Wendell P, et al. Apache spark: a unified engine for big data processing[J]. Communications of the ACM, 2016, 59(11): 56-65.
39. Bachem O F. Sampling for Large-Scale Clustering[D]. ETH Zurich, 2018.
40. Bachem O, ETHZ I N F, Lucic M, et al. Scalable and distributed clustering via lightweight coresets[J]. stat, 2017, 1050: 27.
41. Ackermann M R, Märtens M, Raupach C, et al. StreamKM++: A clustering algorithm for data streams[J]. Journal of Experimental Algorithmics (JEA), 2012, 17: 2.4.
42. Bachem O, Lucic M, Krause A. Practical coreset constructions for machine learning[J]. arXiv preprint arXiv:1703.06476, 2017.
43. Feldman D, Schmidt M, Sohler C. Turning big data into tiny data: Constant-size coresets for K-means, pca and projective clustering[C]//Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, 2013: 1434-1453.
44. Mettu R R, Plaxton C G. Optimal time bounds for approximate clustering[J]. Machine Learning, 2004, 56(1-3): 35-60.
45. Hastings W K. Monte Carlo sampling methods using Markov chains and their applications[J]. 1970.
46. Bachem O, Lucic M, Hassani S H, et al. Approximate K-means++ in sublinear time[C]//Thirtieth AAAI Conference on Artificial Intelligence. 2016.
47. Bachem O, Lucic M, Hassani H, et al. Fast and provably good seedings for K-means[C]//Advances in Neural Information Processing Systems. 2016: 55-63.
48. Ben-David S. A framework for statistical clustering with a constant time approximation algorithms for k-median clustering[C]//International Conference on Computational Learning Theory. Springer, Berlin, Heidelberg, 2004: 415-426.
49. Friggstad Z, Rezapour M, Salavatipour M R. Local search yields a PTAS for k-means in doubling metrics[J]. SIAM Journal on Computing, 2019, 48(2): 452-480.
50. Capó M, Pérez A, Lozano J A. An efficient approximation to the K-means clustering for massive data[J]. Knowledge-Based Systems, 2017, 117: 56-69.