Project 8: Harp MiniBatch Kmeans
Cloud Computing
Spring 2017

Professor Judy Qiu

Goal

The goal for this project is to implement Harp[1] Mini-batch Kmeans from scratch.

Deliverables

Zip your source code and report as username_mbkmeans.zip. Please submit this file to the Canvas Assignments page.

Evaluation

The point total for this project is 6, where the distribution is as follows:

- Completeness of your code (5 points)
- In the report, describe your implementation and the output. (1 points)

You can get up to 4 bonus points based on your extra efforts.

Bonus credits

Some options you may consider to get extra credits:

- Perform experiments on various (small, medium, large, etc) datasets
- Test your algorithm on at least 2 nodes on FutureSystem.
- Implement mini-batch kmeans using other tools/platforms (Spark[2], Flink[3], etc) and compare the performance between different tools/platforms.

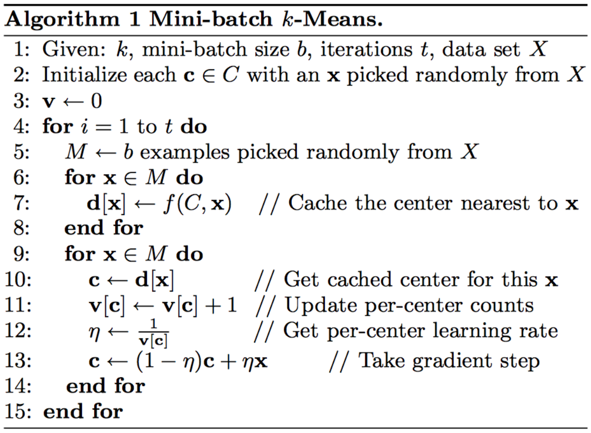
You are encouraged to explore other options to get extra credits. Remember to present all of your extra work in the report.

Dataset

You can implement a script to generate data randomly as your input datasets. You are also free to use public datasets such as RCV1-v2[4].

Mini-batch Kmeans

You can refer to the paper[5] for sequential mini-batch kmeans algorithm. You will need to design how to parallelize the algorithm so that it can run with large scale datasets on distributed computing environment.



Algorithm 1 Mini-batch k-Means.

```
Given: $k$, mini-batch size $b$, iterations $t$, data set $X$
Initialize each $\mathbf{c} \in C$ with an $\mathbf{x}$ picked randomly from $X$
$\mathbf{v} \leftarrow 0$
for $i = 1$ to $t$ do
    $M \leftarrow b$ examples picked randomly from $X$
    for $\mathbf{x} \in M$ do
        $\mathbf{d}[\mathbf{x}] \leftarrow f(C, \mathbf{x})$    // Cache the center nearest to $\mathbf{x}$
    end for
    for $\mathbf{x} \in M$ do
        $\mathbf{c} \leftarrow \mathbf{d}[\mathbf{x}]$    // Get cached center for this $\mathbf{x}$
        $\mathbf{v}[\mathbf{c}] \leftarrow \mathbf{v}[\mathbf{c}] + 1$    // Update per-center counts
        $\eta \leftarrow \frac{1}{\mathbf{v}[\mathbf{c}]}$    // Get per-center learning rate
        $\mathbf{c} \leftarrow (1 - \eta)\mathbf{c} + \eta\mathbf{x}$    // Take gradient step
    end for
end for
```

Figure 1: Mini-batch Kmeans.[5]

References

[1] Indiana University. `https://dsc-spidal.github.io/harp`.

[2] Apache. `http://spark.apache.org`.

[3] Apache. `https://flink.apache.org`.

[4] David D. Lewis. `http://jmlr.csail.mit.edu/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm`.

[5] David Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pages 1177–1178. ACM, 2010.