TUM I 13
Chair of Application and Middleware Systems
Prof. Dr. H.- A. Jacobsen

Tutorial
Distributed Systems (IN2259)

Exercises on Consistent Hashing and MapReduce

Exercise 1 Traditional Hashing vs. Consistent Hashing

A set of N servers (cf. Table 1.2) should be used to cache a collection of M different web sites (cf. Table 1.1) belonging to an application provider. In order to reduce the load on the web server hashing is applied to distribute the web sites among the web caches.
Note: For subtasks (a) and (b) you can neglect the hash values given in the tables.

Name	ID	Hash Value
WebSite_0	0	1A2E
WebSite_1	1	C649
WebSite_2	2	431C
WebSite_3	3	1665
WebSite_4	4	61B3
WebSite_5	5	9271
WebSite_6	6	1CF3
WebSite_7	7	214D
WebSite_8	8	8715
WebSite_9	9	ECA2

Web sites

Name	ID	Hash Value
WebCache_0	0	7912
WebCache_1	1	CAD4
WebCache_2	2	C23E

Web caches

(a) In a first approach, the hash function $h(id) = 7 \cdot \text{id} + 4 \text{ mod N}$ is used to associate web sites with caches. Each page is hashed based on its ID and the result of the hash operation is supposed to determine the ID of the corresponding cache. List the set of web sites that is managed by each cache.

(b) How does the situation change when a new server ('WebCache_3', 3, 2F69) is added to the set of caches? Again, list the allocation of web sites to caches. What do you observe? Quantify the degree of reallocation (i.e., the percentage of web sites that need to be transferred).

(c) In a second approach, consistent hashing is used to associate web sites to caches. Similar to MD5, the hash function that is used for this example produces hex numbers comprised of 4 hex digits. Hence, the range of this function is $[0000, FFFF]$. The hash values for caches and web sites are given in Table 1.2 and Table 1.1, respectively. Based on these values associate web sites to the corresponding caches.

(d) How does the situation change when the new server ('WebCache_3', 3, 2F69) is added to the set of caches? Again, list the allocation of web sites to caches. What do you observe? Quantify the degree of reallocation (i.e., the percentage of web sites that need to be transferred).

(e) Compare the uniformity of the distribution in subtask (c) and (d) by calculating the standard deviation. What do you conclude from the result? The standard deviation should be calculated with the following formula:

$$S = \sqrt{S^2} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2} \qquad \text{, with} \qquad \bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

(f) As we saw, it is possible to have a non-uniform distribution of web sites between caches if there are not enough caches in the system. A possible solution to deal with this problem is to introduce the concept of "virtual nodes", i.e.,

replicas of web caches, where each real cache corresponds to several virtual caches in the circle. Whenever a cache is added, also a number of virtual nodes is created for the new cache, and when a cache is removed, all its virtual nodes are removed from the circle. However, all objects that are handled by a virtual cache are actually handled by the real cache they are associated with.
For the sake of this example the replication factor should be 2 (i.e., for each cache there are two additional virtual caches). The virtual caches are given in Table 1.3.
Give the allocation of web sites to virtual caches and (real) caches and calculate the standard deviation for the real caches. How does the distribution compare to the above scenarios?

Cache	Virtual Cache	Hash Value
WebCache_0	VirtualCache_0_1	8B02
	VirtualCache_0_2	17D1
WebCache_1	VirtualCache_1_1	5CBC
	VirtualCache_1_2	7074
WebCache_2	VirtualCache_2_1	1FA3
	VirtualCache_2_2	8010

Web caches and their virtual nodes.

EXERCISE 2 MapReduce Algorithms

Formulate the following algorithms for MapReduce. Explain how the input is mapped into (*key, value*) pairs by the map stage, i.e., specify what is the key and what is the associated value in each pair and how the key(s) and value(s) are computed. Also explain how the (key, value) pairs produced by the map stage are processed by the reduce stage to get the final result. Please use the below scheme:

MAP: <What the map function does>

Input: <define input>

Output: <define output>

REDUCE: <What the reduce function does>

Input: <define input>

Output: <define output>

(a) Word count: Count the frequency of word apperances in a set of documents.

(b) Search for a pattern: Data is a set of files containing lines of text. Output the file names that contain this pattern.

(c) Sorting: Given a set of files, one value per line, sort the values. Assume that all values are unique.

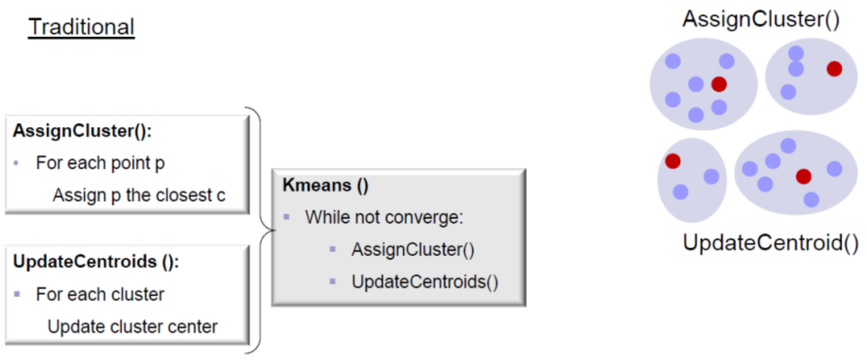
EXERCISE 3 K-means in MapReduce

K-means clustering aims to partition n observations into K clusters in which each observation belongs to the cluster with the nearest mean. The basic procedure for K-means clustering is:

- Partition $\{x_1, \ldots, x_n\}$ into K clusters, where K is predefined.
- Initialization: Specify the initial cluster centers (centroids), i.e., K.

- Iteration until no change:
  - For each object x_i
    - Calculate the distance between x_i and the K centroids
    - (Re)assign x_i to the cluster whose centroid is closest to x_i
  - Update centroids based on current assignment

We want to develop the MapReduce formulation for K-means clustering. Bellow you can see a traditional implementation of K-means, where p is the observation x_i and c is the centroid.



(a) The MapReduce formulation of K-means needs a driver or wrapper around the normal execution framework. What is the reason for this?

(b) The MapReduce K-means algorithm can be formulated using the following components:

- Driver or wrapper
- Mapper
- Reducer

Consider that a single file contains the predefined cluster centers K and the data points are distributed in several files. Provide a short description defining the task performed by each component of MapReduce K-means and define their input and output.

(c) What characteristic of K-means could cause a large computation overhead?