CS898D – Data Mining

Homework #1

Chakradhar Reddy Donuri

E949F496

Q1. Discuss statistical limits of Data Mining?

Answer:

1. Total Information Awareness (TIA)

- TIA is a project which was planned to mine all the data it could find, including credit-card receipts, hotel records, travel data, and many other kinds of information in order to track terrorist activity.
- The prospect of TIA raised many technical questions about its feasibility and the realism of its assumptions.
- The major concern is that when we use such system we may find activities that look like terrorist behavior but on the other hand you may find illicit activities that are not terrorism.
- Such activity may result in adverse situation when people dig into the private data and information about the individual is no more secure.

2. Bonferroni's Principle

- Consider any amount of data we can expect events of this type i.e. malicious to occur, even if the data is completely random and such occurrences increase with increase in the amount of data.
- Such occurrences are known as "Bogus"
- To overcome such problem we have "Bonferroni's Principle" which avoids bogus occurrences.
- It calculates the expected number of occurrences of the events you are looking for, on the assumption that data is random. If this number is significantly larger than the number of real instances you hope to find, then you must expect almost anything you find to be bogus.
- Bonferroni's principle says that we may only detect terrorists by looking for events that are so rare that they are unlikely to occur in random data.

3. **It violates user privacy:** Data mining collects information about people using some market-based techniques and information technology. And these data

mining process involves several numbers of factors but while involving those factors, data mining system violates the privacy of its user.

4. **Misuse of Information**: The possibility of safety and security measure are really minimal. And that is why some can misuse this information to harm others in their own way. Therefore, the data mining system needs to change its course of working so that it can reduce the ratio of misuse of information through the mining process.

5. **Accuracy of the Data**: Most of the time while collecting information about certain elements one used to seek help from their clients, but nowadays everything has changed. And now the process of information collection made things easy with the mining technology and their methods. One of the most possible limitations of this data mining system is that it can provide accuracy of data with its own limits.

Q2. Explain Distributed File System.

Answer:

- Distributed File System (DFS) is often used for Large Scale file-system organizations.
- DFS is effective when files are enormous, possibly a terabyte in size. If you have only small files, there is no point using a DFS for them.
- DFS is effective when files are rarely updated, they are read as data for some calculation, and possibly additional data is appended to files from time to time. For example, an airline reservation system would not be suitable for a DFS, even if the data were very large, because the data is changed so frequently.
- In DFS implementation files are divided into chunks, which are typically 64 megabytes in size.
- These Chunks are replicated, perhaps three times, at three different compute nodes. Moreover, the nodes holding copies of one chunk should be located on different racks, so we don’t lose all copies due to a rack failure
- Users can decide the chunk size and the degree of replication.
- To find the chunks of a file, there is another small file called the master node or name node for that file. The master node is itself replicated, and a directory for the file system as a whole knows where to find its copies
- The directory itself can be replicated, and all participants using the DFS know where the directory copies are.
- There are several distributed file systems :

1. The Google File System (GFS).
2. Hadoop Distributed File System (HDFS), an open-source DFS used with Hadoop
3. CloudStore, an open-source DFS originally developed by Kosmix.

Q3 What is MapReduce? Illustrate the working of MapReduce with a simple illustration.

Answer:

MapReduce:

- MapReduce is a style of computing that has been implemented in several systems, including Google's internal implementation (simply called MapReduce) and the popular open-source implementation Hadoop which can be obtained, along with the HDFS file system from the Apache Foundation.
- MapReduce is used to manage many large-scale computations in a way that is tolerant of hardware faults with two functions called MAP and REDUCE.

Working of Map-Reduce

Map reduce computation executes as follows:

1. Some number of Map tasks each are given one or more chunks from a distributed file system. These map tasks turn the chunk into a sequence of key-value pairs. The way key-value pairs are produced from the input data is determined by the code written by the user for the Map function.
2. The key-value pairs from each Map task are collected by a master controller and sorted by key. The keys are divided among all the Reduce tasks, so all key-value pairs with the same key wind up at the same Reduce task.
3. The Reduce tasks work on one key at a time and combine all the values associated with that key in some way. The manner of combination of values is determined by the code written by the user for the Reduce function.
   - Map :- Extracts the data in which the user is concerned then group by Key i.e. sort and shuffle
   - Reduce :- Aggregate, summarize, filter or transform

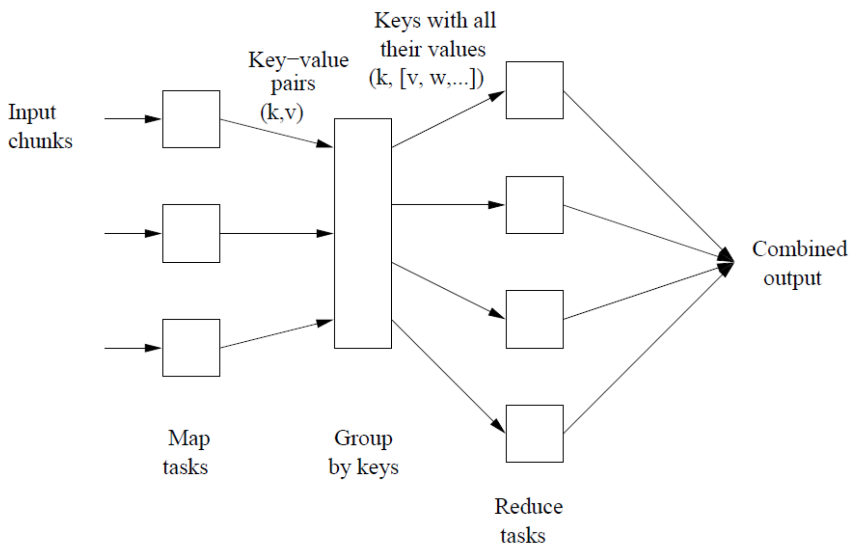
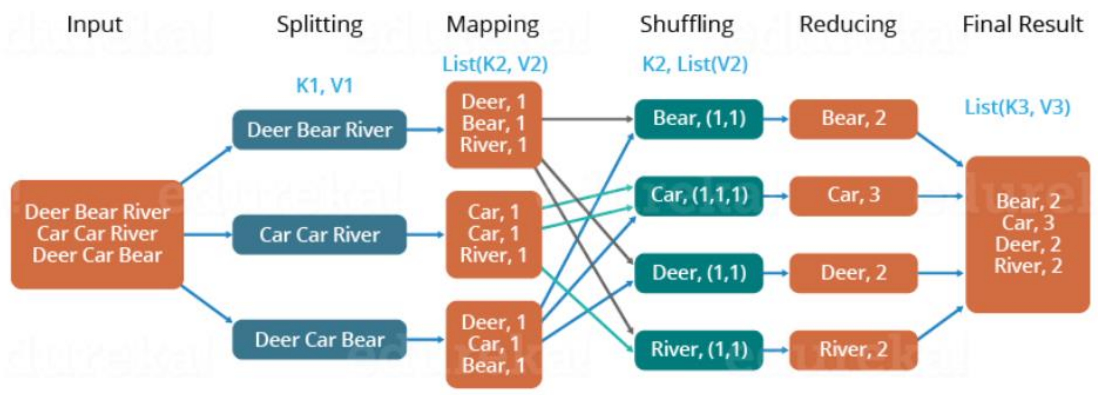


Figure 2.2: Schematic of a MapReduce computation

Word count Example of Map-Reduce

Dear, Bear, River, Car, Car, River, Deer, Car and Bear



- First, we divide the input in three splits as shown in the figure. This will distribute the work among all the map nodes.
- Then, we tokenize the words in each of the mapper and give a hardcoded value (1) to each of the tokens or words. The rationale behind giving a hardcoded value equal to 1 is that every word, in itself, will occur once.
- Now, a list of key-value pair will be created where the key is nothing but the individual words and value is one. So, for the first line (Dear Bear River) we have 3 key-value pairs – Dear, 1; Bear, 1; River, 1. The mapping process remains the same on all the nodes.

- After mapper phase, a partition process takes place where sorting and shuffling happens so that all the tuples with the same key are sent to the corresponding reducer.
- So, after the sorting and shuffling phase, each reducer will have a unique key and a list of values corresponding to that very key. For example, Bear, [1,1]; Car, [1,1,1].. etc.
- Now, each Reducer counts the values which are present in that list of values. As shown in the figure, reducer gets a list of values which is [1,1] for the key Bear. Then, it counts the number of ones in the very list and gives the final output as – Bear, 2.
- Finally, all the output key/value pairs are then collected and written in the output file.

Q4 Explain the costs associated to Mapper and Reducer and their trade-off.

Answer: For a Map-Reduce Algorithm we have:

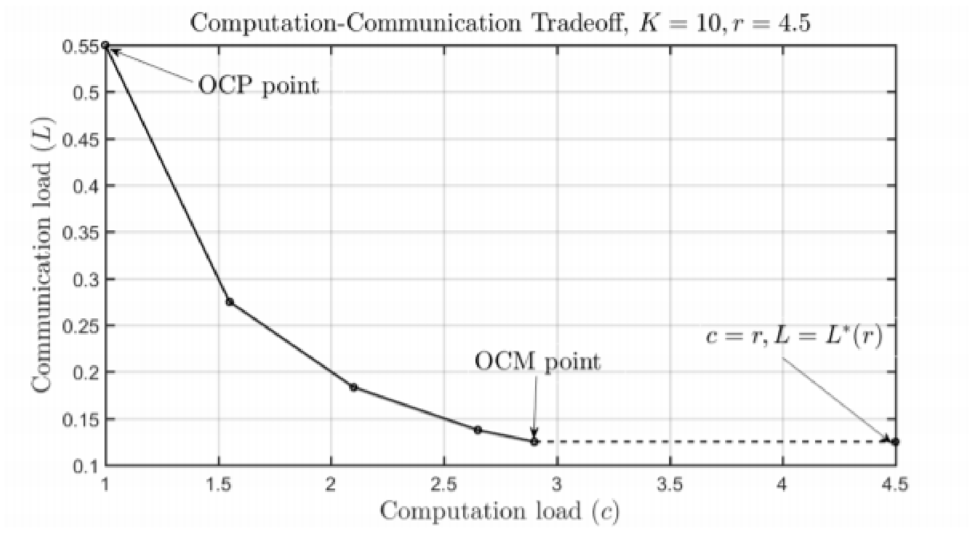
1. ***Communication cost*** = total I/O of all processes **=** input file size + 2 $\times$ (sum of the sizes of all files passed from Map processes to Reduce processes) + the sum of the output sizes of the Reduce processes.
   Therefore, Total Communication cost = $O(|R|+|S|+|R \bowtie S|)$
2. ***Elapsed communication cost*** = max of I/O along any path. It is the sum of the largest input + output for any map process, plus the same for any reduce process.
   1. We're going to pick ***k*** and the number of Map processes so that the I/O limit ***s*** is respected
   2. We put a limit ***s*** on the amount of input or output that any one process can have. ***s* could be:**
      1. What fits in main memory
      2. What fits on local disk
3. **(*Elapsed*) *computation cost*** analogous but count only running time of processes.

   Below figure depicts the Communication and computation trade-off with fixed storage load r=4.5 and K=10

$$L*(r) = 1/r * (1 - r/K)$$

   OCP - optimal computation curve

   OCM - optimal Communication curve



Q5 What are the challenges associated to cluster computing? How are they mitigated?

Answer:

The challenges of cluster computing are as follows:

1) Middleware:
   To produce software environments that provides an illusion of a single system image, rather than a collection of independent computers.
2) Program:
   The applications that run on the clusters must be explicitly written which incorporates the division of tasks between nodes, also the communication between them should be taken care of.
3) Elasticity:
   The variance in real-time response time when the number of service requests changes dramatically.
4) Scalability:
   To meet the additional requirements of a resource thus effecting the performance of the system

6) Network considerations:

The manager/worker architecture clearly requires a reliable network between the components. It must have high enough bandwidth in order to allow the downloads of any large database files required to succeed in a reasonable amount of time, and be able to sustain the large flow of connections from the workers to the manager.

7) Managing complexity:

Conceptually, each of the distributed computation management systems is quite simple. A manager maintains state of work units, and workers are delivered work upon request, compute the results for that work, and return the results

8) Programmability Issues:

This might be the case if the components are different in terms of software from each other, and then there may be issues when combining all of them together as a single entity.

Avoid Network Overhead-

To reduce this network overhead, we apply network coding only at particular nodes to reduce the number of coding nodes. This network coding can be performed at cluster heads. To identify these cluster heads, we are using the clustered based routing protocol (CBRP). We apply network coding at cluster heads to reduce the coding overhead and reduce redundant transmissions and therefore reduce the number of transmissions. Therefore, it reduces the energy consumption involved in sending and receiving packet transmissions

Store Data Persistently:

The ability to separate compute and storage allows database software increased availability and scalability.

"Separating compute and storage", involves designing databases systems such that all persistent data is stored on remote, network attached storage. Local storage is only used for transient data, which can always be rebuilt (if necessary) from the data on persistent storage.