Abhinav Antani
192CS002
Distributed Data Management(CS750)
MTech(CSE) Sem2
9426207062
antaniabhinav@gmail.com

Title: Range Query optimization in distributed data

Abstract:

Query is question asked on data. A single query can give multiple records as results which satisfy given criteria. A Range query is a query which wants results to have some entity value within some range. In every Domain, data generated is increasing exponentially day by day. Now in real life this large volume of data is distributed at various places. If range queries come for this large volume of data then for each record we need to see whether it satisfies the given criteria, if so then return those records. When large volume of distribute data is considered this task is very time consuming. This process can be optimized and time consumed can be reduced. In this paper I have used the metadata of the distributed data to optimize the time. Using this technique query will be asked only to certain databases and not whole data. Because of this time will be saved. Here in this experiment I have made range query optimization for numerical data.

Key Terms: Range Queries, Distributed Data, Meta-Data

Introduction:

In every domain data is increasing exponentially day by day. With this large volume of data it becomes necessary to keep the query processing in a distributed data system. This query can be a range query on the data as well. Range Query is database operation that fetches all the records where some attribute value is within some lower bound and upper bound.
Now when a large system is checked for fetching the records it takes a significant amount of time. This time can be improved and overall system performance can be optimised by optimizing range query operations. In this work I have optimized the performance of range query for numerical data. To optimize the performance I have used Meta data of the distributed data system.

Meta data is the data about data. In distributed data systems where data is in large amount meta data is very important while discovering, accessing and retrieving the data. Meta-Data gives us the information about data, such as location, topic, description, content, structure etc. Here in this work I have used the sub category "content" meta data to optimize the range queries operations. In this work I have used minimum and maximum value of the attribute to decide whether to search within the data store or not. It helps to discard some set of data as the probability of finding answer is near to null in such search scenarios.

Applications of range query on numerical data are many. If we talk about the internet search every webpage is given some rank based on the document and key word matching. Ranks are in numerical form. Example as, if we want to show the documents, which have ranks in first

500 then these range of ranks search can be optimised in the internet search and response time can become low. In the domain of Platforms such as YouTube can have advantage of this range query optimization. If some company wants to give money to its advertisers by some range of likes and views then this likes and views can be searched optimistically by this range query optimizer. The range query can optimize business applications for retail. Store managers can drastically enhance their search results and derive greater insights leading to better user experience and business results.

This work optimizes the performance of distributed data system. Range operations on numerical data are optimized using metadata. Example as, if there was no range query optimization done and time taken to execute x operation is 0.32 seconds and after applying this range query optimization same x operation shall take 0.23 seconds of time.

Literature Review:

Different systems have been proposed to implement range queries on the data. Optimization in these systems is achieved by adding auxiliary space or by storing data in some data structure. For storing the location to know where data is stored data structures like Hash Table, Prefix Hash Tree, Skip Graphs excreta can be used. Later by applying same hash function to the queried data we can know where to search that data.

Some proposed the use of Prefix Hash trees coupled with distributed hash tables. They did range query optimization for alphabetic data. They wanted to search a word in the dictionary in some given range. Example as, retrieve all the words which starts with 'ac'. To achieve this they proposed to break the search in various sub ranges like if we want to have words starting with 'ac' then search for the words starting from 'aca' , 'acb' , 'acc' …. then give all the possible results.

Some proposed the use of Skip Graphs. Skip Graphs rely on the membership vector and they are tracked using Distributed Hash Table. This Distributed Hash Table is built on physical nodes and these physical nodes have pointer pointing from one point to another depending on how they are connected.

Proposed Work

In this work I am optimising the performance of the distributed data system for range queries. These range queries are in particular for numerical types of data. Many applications uses distributed data systems and have many queries to answer at the same time. It becomes necessary for the application to compute the results very fast and show it to the user. When this data is small it can be captured in a single machine. Optimizing such systems is easy by making that particular machine fast. But when this data is distributed over large network this task becomes tedious. For this intensive amount of data we need to improve the performance of the query processor. Range queries are part of these query processors. In a stand-alone system range queries are not optimized. In such system for each record we need to check whether it satisfies the given criteria or not. In our scenario this criteria is range from lower bound to upper bound. In range query for every query we ask 2 queries. Example as, assume that we want to search for the records whose value is within the range of 250 to 500. This query is divided in two different non range query that is attributes with value grater than 250 AND less than 500. Records satisfying both of these conditions (AND Operation) will be our

desired result. Imagine doing this task for a huge amount of data. This would take significant amount of time. To improve this situation following work proposes a range query optimizer which reduces the amount of time taken to retrieve the suitable data within the given range. I am doing this optimization for the numerical data. It is chosen because in the base paper which I selected has done this for Alphabetical data using different technique. The technique which I have used involves the use of meta data of the distributed system. From this work I expect to reduce the time taken to execute a range query over a distributed system.

Important and major concept of distributed data system, which is used here is Meta Data. Using the information about the data I am deciding whether to search in particular set of records or not. Meta data captures and shows us the information about the data. That's why it is often called as the data of data. This data contains information such as description of the data: what type of data is stored, structure: description of the structure of the data, Location: where data is stored, content: which kind of data is stored etc. This description, location, content, structure of the data helps the system to discover, access and later if necessary retrieve the data. In large distributed data systems meta data helps to organize, and characterize the data. Organizing and characterizing the data helps in gaining the knowledge about the data. After having the enough knowledge about the data, we can explore and use some background information about the data for our own purpose. Repeatability, Reproducibility and transparency are good qualities of the research. In data intensive systems, in order to provide Repeatability, Reproducibility and transparency, it is important to provide automatic information about the data without digging in the data. With the help of meta data we can achieve that. This can be understood by following example, consider that you want data about the records of cricketer Sachin Tendulkar and you have the data of current players. After digging through all the records you will come to know that data for Sachin Tendulkar is not present in the data that you have. Life would have been easier for you if beforehand you knew that this data was of current players and it does not contain the data of retired players. This kind of information about the data is present in the meta data. We can save our time by not looking for something which is not there by using meta data.

Goal for this work is to optimize the performance of range query, that is reducing the time taken to compute the results for a range query in distributed system. Assume that you want to search for all the records which satisfy an attribute value within a given range of x to y. Now assume that a stand alone system takes t1 seconds and my proposed system takes t2 seconds to compute the results. The goal is to make time taken by the proposed system which is t2 to be less than time taken by the stand alone system which is t1. In order to match the results with the goal I have used the concepts of Meta Data. Now important question is why meta data? Answer is, to optimize the performance meta data can be used in the data intensive applications to discover, access and later if necessary retrieve the data. This work is carried out to optimize the range query operation specifically on numerical kind of data. Assume that you have 1 lakh data records and you want to search for the data which are in the specified range of x to y. Our distributed system is scattered in 10 different locations and all the location have the data about the application. All of this 10 location have equal amount of data. Which implies that each location has 10 thousand of records. In a traditional stand alone distributed data system, if this range query would have been executed then this query would have been broadcasted in each of these 10 locations and our result would have been computed. This means that for a single range query 1 Lakh of records would have been queried twice (as explained earlier, for upper and lower limit). In proposed system it will be done in less than that. Using Meta data we will get the minimum and maximum value of queried attribute in each of the 10 locations. Now a location will be checked if and only if our range x and y is within the minimum or maximum limit. Because if x or y are not in the range

of minimum to maximum in that location then it becomes void to search in that location as there will not be any record which satisfies the given range x to y. The proposed system works on this main principle. So if there were only 2 locations out of 10 which would have satisfied the conditions, then only those 2 locations will further be searched. So by applying the information which was there in meta data we are now able to get the same result by searching only 20 thousand records instead of 1 lakh records. This gives us major improvement in the time taken.

For any system to work properly there has to be some method by which they are needed to do the task. Some architecture is followed by every system. This particular system follows master slave architecture. Assume that in master slave architecture, system is not having any optimizations on how a range query will be processed. As data is gigantic it will be stored in a distributed network. The master will have all the information about it's various slaves which are holding the data for the application. Let's say there are 10 sites which are holding data for the distributed system. Now, if a range query comes then that range query will come first to the master. Then master will broadcast this query to each of the 10 sites to get the results. After receiving the query each site will execute it and will prepare the results. Then these results will be given to the master and it will show it to the user. Now here many of the sites might be doing unnecessary work. Here is how system will work after the optimization. Range Query will come to the query processor and master will see that query has been asked. After the optimization using meta data master will know the content of each of the sites. Here this content is nothing but minimum and maximum value of the attribute in that particular site. So master will have minimum and maximum value of each site. When the range query will come it will see the range asked. Now using the minimum and maximum value of each site and this queried range value, the master will compute which site to query. Now let say master computes that only 3 sites might have the queried data, then only those 3 sites will compute the results. Which implies instead of 10 now only 3 sites will be queried. This will improve the time taken to solve the range query and it will optimize the performance.

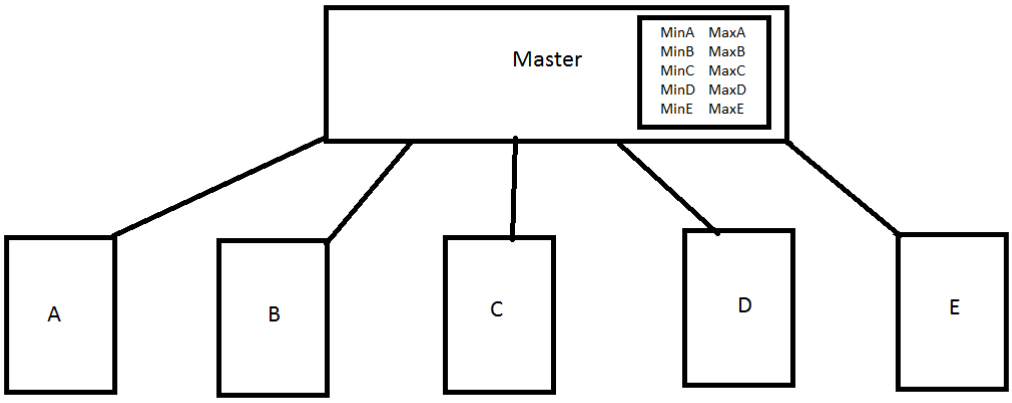

Figure 1: Working Diagram

Above is the block diagram of the system. As you can see master is having information about all the sites. This information is minimum and maximum value of the attribute. Here minA

and maxA represents minimum and maximum value of the attribute for which query is asked. These values are pre computed and are stored in some variables at the master's location. So that each time we do not have to compute them. Now question arises what if data is altered at the sites? If that happens then in constant time variables such as minA and maxA can change the value. Example as, let say minA and maxA are having values as 0 and 100 respectively. Now due to some reason the site A has to alter some data points. Now at the time of altering the data points with new values, if those new values can change the minimum and maximum values then we can change the minA and maxA directly. For example if one of the new value is 150, then this 150 will be compared with the minA which is 0, then still minA remains 0, now it is compared with maxA. After comparing maxA will become 150. And this is how in constant time min and max values can be updated if data is altered. Now question arises what if bunch of data is altered? Then for those bunch of data min and max will be computed. With this new min and max value minA and maxA is compared and minA and maxA are updated accordingly. This is how the table is maintained in the master. That is ultimately used in the range query optimization.

Now how would a computer do this task? For this we need to understand the algorithmic working of this model. We will understand it by using some example. Here as explained earlier in the above paragraph we have 5 sites named as A,B,C,D,E. One master program is there which guides us on where to search. From master we will know which sites will execute the query and will provide us the results. Master has the copy of below Meta data table.

Site	Minimum Value	Maximum Value
A	0	1100
B	1300	1564
C	1600	1750
D	2200	2350
E	2400	3000

Table 1: Meta Data Table

There are three major columns in this table such as site, Minimum Value, Maximum Value. Here these minimum and maximum values are for the attribute for which range query optimization is being done. Now let's see how this works. Assume that master receives the range query in the range of 1250 to 1650. Now master will use one equation to check whether a site has records in this range or not.

$$min\_range < max\_X \text{ and } max\_range > min\_X$$

Equation 1: Condition check of site

Above is the equation used by master to check whether a site X should be searched or not. Here min_range and max_range is the range given in the query where as max_X and min_X

are the values of minimum and maximum value at site X. Following the above example min_range is 1250 and max_range is 1650. Now initially status of every site is false, it means that no site will be searched. A site will be searched only if it satisfies the 'equation 1'. Now for site A min_A = 0 and max_A=1100. Now as 1250<1100 (min_range < max_X) will be false. Like wise 1650 > 0 (max_range > min_X) will be true. And ultimately whole equation will give us false (False & True = False) as the result. That means site A will not be checked for the records. Now if we do it for site B, it will be True. Because, 1250<1564 (min_range < max_X) will be true , and 1650 > 1300 (max_range > min_X) will also be true. Eventually making 'equation 1' true for site B. Likewise if we compute the equation value for C,D,E the answer will be True, False and False respectively. And now as only B and C site are true , only B and C site will execute the query and time will be saved for the sites A, D, and E. This is how this system works and optimizes the performance of range queries.

To make this system python was used as the programming language. To make the database of different sites (like A,B,C,D,E) xampp is used. Xampp is an open source platform to make local host, databases etc. Here using sql various databases are maintained. To connect python with the database 'mysql' library of python is used. Mysql library is also an open source which can be downloaded using the command 'pip install mysql' in the command line shell of python. The reason why python is chosen as the language of implementation is that it contains various libraries using which data can be handled easily and without further computations.

As this experiment was to optimize range query performance of the system on a numerical data , YouTube's data set is used. This data set was downloaded from the site named as www.kaggle.com the dataset containes information such as video Id , name of the video , views , likes , no of comments , genre , channel name , etc. It contains information of various locations such as Australia , America ,Europe . From this data set 'likes' column is chosen. This is the attribute where range query optimization will be applied. To make things easy all the other columns are neglected and only name of the video and no of likes are considered. Here likes column is chosen because of two major reasons. First one being that it is numerical type of data. Second reason is that many times range query can be asked by the analyst on the 'likes' column to improve their chances of putting advertisement on some value based on the popularity.

In this work I have worked on Range queries on distributed data and how one can improvise it. Using the information about data which is also called as Meta Data I am able to improvise the system. It is achieved by keeping the updates of minimum and maximum value in the master. We can use this information later and improvise the performance of the system when range query is asked.

Now the system is made but how to evaluate whether it is working or not? Is it really optimizing the performance of the system? To do that various experiments were done on the YouTube dataset and in each of the experiments optimized version was taking lesser time than the stand alone system. To evaluate, some query X was run on the stand alone system (without optimization) and time taken was recorded. Then same query X was passed to the

system proposed (Optimized) and time taken was recorded. From the time taken by both the system it is clear that system proposed is improving the results of the query processor. Below is the table of subset of experiments done.

Range	Time Taken: Stand Alone	Time Taken Optimized	Speed Up
1550-7826	0.28	0.18	0.64
50-1926	0.19	0.11	0.57
10000-21000	0.15	0.10	0.66
15000-38000	0.18	0.12	0.66
10000-100000	0.32	0.23	0.71

Table 2: Results

From the above table we can see that in each query some sort of speedup is achieved. This shows that system is optimizing the performance of the range queries done on the attribute 'likes'. And average speedup achieved by the new system is 0.65.

Conclusion:

This work was done to optimize the performance of range query over distributed data. Mainly the information about the data was used to optimize the performance. Information about data is also called as meta data. Minimum and maximum value in the particular site for the range query attribute was used to optimize the performance. With the help of this minimum and maximum value it is decided whether to search on that site or not. By using this optimization speed up of 0.65 was achieved in the system. This experiment was done on the data set of YouTube, particularly on the 'likes' attribute. This data set was downloaded from an open source website named www.kaggle.com.

Here in this experiment each site is individual to each other. Means no site is dependent on each other on how to compute the results. Parallelism of the system can be done as future work. Many sites can work to gather and compute and send the result simultaneously to the master and master can show the results. This will further improve the performance. For example if site A, B , C are searched after the master says then currently time taken is summation of time taken by the sites A , B, C. But if this work is done in parallel as they are independent to each other same query will take maximum of time taken by A,B or C. This will further improve the performance of the system as summation of time taken is always grater than maximum of time taken

References:

[1] D. Karger, E. Lehman, T. Leighton, R. Panigraphy, M. Levine and D. Lewind, "Consistet hashing and random trees: distributed caching protocols for relieving hot spots on the world wide web.," In Proceedings of the twnty-ninth annual ACM symposium on Theory of computing., pp. 654-663, 1997

[2] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek and H. Balakrishnam, "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications," IEEE/ACM Transactions on Networking, vol. 11, no. 1, pp. 17-32, 2003.

[3] P. Yalagandula, "Solving Range Queries in a Distributed System," Tech. Rep. , no. TR-04-18, pp. 17, 2004.

[4] S. Ratnasamy, J. M. Hellerstein and S. Shenker, "Range Queries over DHTs," IRB-TR-03-009,vol. 03, no. 009, pp. 1-2, June 2003.

[5] Richard Price , Lakshmisinh Ramaswamy , Seyedamin Pouriyeh , " Efficient Processing of Range Queries over distributed relational databases"